ESC-TR-92-159

Technical Report
974

Overview of Enhanced Data Stream Array Processor (EDSAP)

A.G. Rocco
P.D. Linton
J.K. Noonan

6 May 1993

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS

Prepared for the Air Force under Contract F19628-90-C-0002.

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Air Force under Contract F19628-90-C-0002.

The ESC Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Gary L. Tutungian, Administrative Contracting Officer,
Directorate of Contracted Support Management

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

OVERVIEW OF ENHANCED DATA STREAM ARRAY PROCESSOR (EDSAP)

A.G. ROCCO
P.D. LINTON
J.K. NOONAN
Group 47

TECHNICAL REPORT 974

6 MAY 1993

Approved for public release; distribution is unlimited.

LEXINGTON MASSACHUSETTS

EXECUTIVE SUMMARY

The Enhanced Data Stream Array Processor (EDSAP) is a scalable version of the DSAP processing architecture [1] that will allow massively parallel processing. Lincoln Laboratory previously designed two generations of DSAP processors; these programmable signal/data processors provide very high throughput in small, lightweight packages that require low power use. The EDSAP design builds upon the DSAP architecture in a way that provides for processing in the gigaop (billion operations per second) to teraop (trillion operations per second) range. This capability is being accomplished primarily by a significant redesign of high-speed I/O and interprocessor communication. Other architectural improvements, as well as rapid technological advances, contribute to the improved processor. This report provides an overview of the architecture of the EDSAP. The first EDSAP board is expected to be available in late 1993.

The first-generation DSAP processor was a subsystem of ground-based demonstration radars built by the U.S. Army's Harry Diamond Laboratories for ground surveillance of moving tactical vehicles. Two NMOS, application-specific integrated circuits (ASICs) were designed to perform the high-speed signal processing. The processor weighs 50 lbs, dissipates 300 W, and occupies 1.25 ft^3. The signal processing portion (36 MOPS) can perform moving-target detection on 128 range cells at a pulse repetition frequency (PRF) of 4000 Hz[1] (512,000 samples/second).

The second-generation processor was designed for a lightweight airborne radar carried by a small unmanned air vehicle. The processor successfully performed real-time signal and data processing and radar control as part of the Unmanned Air Vehicle (UAV) radar [2]. During the course of the UAV program, the NMOS custom chips were redesigned in a 1.25-μm double-metal CMOS technology. The signal processing portion of this processor (360 MOPS) was no longer the limiting factor in determining the system's wide-area surveillance capability. The processor easily performs moving target detection and classification from the airborne platform on 480 range gates at a PRF of 7000 Hz. The processor weighs 55 lbs, requires 265 W, and occupies 1.6 ft^3. The processor could also support a real-time, low-resolution (3 m), 2-km-wide stripmap synthetic aperture radar (SAR) mode for detecting stationary ground targets. Other programmatic considerations prevented the SAR mode from being implemented.

The EDSAP is being designed to perform onboard real-time processing to form high-resolution SAR images and to detect and classify stationary ground targets. The significant advance of this new-generation processor is a high-speed, packet-switched, bit-serial network; this network provides high-throughput interconnection of a large number of processors and I/O channels using a manageable number of interconnect wires. The baseline EDSAP processor described in this document is targeted at real-time processing for the Lincoln Laboratory 35-GHz SAR system. Larger data storage (more memory per processing element than in the previous-generation processors), greater precision (24-bit arithmetic instead of 16-bit), and more than 10 GOPS of processing capability will be available to create high-resolution (1 ft) fully polarimetric SAR images and to detect and classify stationary tactical ground targets or strategic relocatable targets in real time. This SAR application is described in Appendix A. Currently, the baseline processor is expected to weigh 70 lbs and require 1 KW in a 3 ft^3 package.

---

1. The radar actually operated at a PRF of 6000 Hz, but only 64 out of every 96 pulses were processed.

The ability to scale the EDSAP has made feasible a processor that could perform the SAR processing described above in a high-altitude UAV platform at a rate of 75 km^2/min. The 672-processing-element (67 GOPS) EDSAP would be fabricated using multichip module (MCM) technology to meet the size, weight, and power constraints of the platform.

Although the EDSAP is being developed for radar signal processing, it is a general-purpose array processor. Its architecture will make it applicable to a wide range of real-time signal processing tasks, particularly those where electrical power and weight are at a premium, yet programmability is desirable. The architecture is flexible and provides easy interfacing to commercially available components. Its flexibility is demonstrated in Appendix B, which describes an EDSAP-based processor design for an application that includes adaptive beamforming.

The EDSAP builds considerably on the existing proven architecture and software tools of the DSAP. The DSAP software development environment includes (1) a debugger, (2) a functional simulator that allows a programmer to develop and test code in a friendlier environment than that of the actual hardware, and (3) an assembly-level programming language. Most signal processing algorithms currently running on the DSAP will be able to run on the EDSAP with few or no changes.

The following design goals will be met by the EDSAP:

1. meet the requirements for proposed SAR and other radars;
2. provide the ability to interface to other high-speed components (e.g., I/O devices and DSPs) and provide an interface to an industry standard (VME) bus for low-speed data, control, and diagnostics;
3. be scalable and versatile enough to meet the requirements of many other systems. Using basic building blocks, it will be practical to configure systems with capabilities ranging from a few hundred MOPS to more than a teraop. The EDSAP will allow evolutionary development, incorporating new technology as it becomes available.

ACKNOWLEDGMENTS

The processor described in this document, the EDSAP (Enhanced Data Stream Array Processor), is the next-generation successor to the DSAP processor that was designed and built at Lincoln Laboratory during the middle and late 1980s. The authors would like to thank C. Edward Schwartz and Gerald B. Morse for their continued leadership, management support, and guidance throughout this period. The authors would like to acknowledge F. Edward Hall and Quentin L. Klein, who, along with A. Gregory Rocco, were the principal architects of the DSAP. Our thanks to Donald Malpass for his work on the DSAP as well as for proofreading this document and providing many helpful comments.

We would like to acknowledge Gary A. Shaw, chairman of a Lincoln Laboratory-wide radar signal processor study. Two of the authors served on this study, which encouraged us to design the EDSAP with an open architecture. In particular, the study-related work helped crystallize the idea of developing the EDSAP PE as a programmable interface chip. Some of the wording in Appendix B was taken directly from the final report of this study [3]. We would like to thank David L. Briggs, who commissioned the study.

Finally, we would like to thank Steven M. Auerbach, our technical editor, for the many helpful editing sessions he has spent with each of us.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

The processor described in this document, the Enhanced Data Stream Array Processor (EDSAP), is the next-generation successor to the DSAP processor that was designed and built at Lincoln Laboratory during the middle and late 1980s. Both the DSAP and the EDSAP are programmable, compact signal and data processors. Their architectures consist of multiple independently programmable array processing elements; this approach combines complete programmability with the benefits of parallelism and pipelining inherent in array operations. Although radar applications have driven both processors' development, they are general-purpose fixed-point array processors; thus they are applicable to a wide variety of real-time signal processing tasks for which low power use and low weight are at a premium and programmability is desirable. The DSAP has provided real-time processing for a number of radar systems. Most notably, it was a key subsystem of the Unmanned Air Vehicle (UAV) radar [2], performing moving-target detection as well as classification of vehicles as wheeled or tracked. The processor also served as the UAV radar controller, providing an interface to the radar front end.

The EDSAP retains all of the desirable features of its predecessor and will meet processing requirements for the 1990s and beyond. While advances in fabrication techniques alone would allow the EDSAP to be significantly more capable than the DSAP, the architecture has also been improved to enhance the processor's scalability.

The importance of scalability can be understood by briefly describing a sizing exercise that used the DSAP architecture. All of the DSAP's processing elements (PEs) receive and transmit data using a single high-speed parallel data bus. Attempts to increase the processing capability of the DSAP by increasing the number of PEs were stymied by this single-bus limitation. Depending upon the application, the bus can support about 12 PEs. Ultimately, designs were proposed in which processing capability was increased by effectively making multiple copies of the DSAP architecture. For example, if 36 PEs were required for a synthetic aperture radar (SAR) application, the design used three "clusters" of processors. Each cluster contained approximately 12 PEs and its own high-speed data bus; communication between clusters was realized in an ad-hoc manner. The resulting architecture was cumbersome and imposed severe constraints upon interprocessor communication across clusters. The EDSAP architecture alleviates the scaling problem by replacing the single parallel bus with a packet-switched bit-serial network of high-speed I/O communication channels. The number of communication channels can easily be increased to accommodate the required number of processing elements, allowing for a massively parallel computing architecture. In addition, each processing element has enough resources to minimize the amount of communication (i.e., each has enough local memory that it does not have to share memory with a neighbor).

Figure 1-1 is a block diagram of the proposed EDSAP baseline system. In the present design, 21 processing boards contain six PEs each. Another board serves as the VME interface. There are 32 network ports available for high-speed I/O (for example, for radar data input). The 126-PE system depicted in the figure is designed to perform real-time SAR processing, including detection and identification, as part of the Lincoln Laboratory 35-GHz SAR radar. The processing includes forming 1-ft-resolution images in each of three polarizations. The stripmap swath that can be processed is 470 m wide, providing coverage at a rate of 2.8 km^2/min. It is estimated that the 35-GHz SAR processing will require 37 PEs. Appendix A discusses the processing sizing operation in some detail.

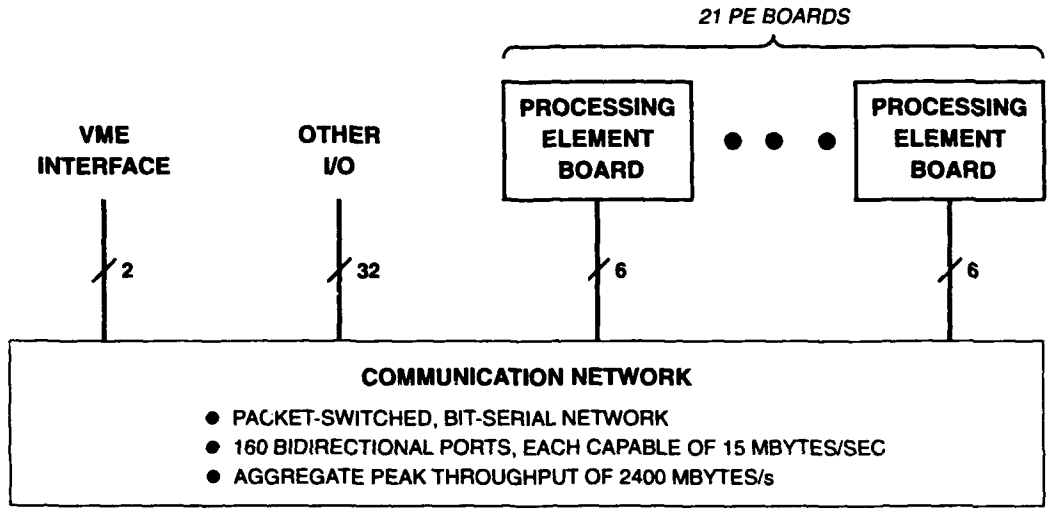

Figure 1-1. Baseline EDSAP for Lincoln Laboratory 35-GHz SAR application.

The backplane contains a communication network that will operate (conservatively) at 200 Mbaud, with the ability to maintain an average transfer rate of 15 Mbytes/sec. Each PE board has six communication ports connected to the backplane.

As indicated in Figure 1-1, 160 bidirectional ports provide access to the communication network. The network can sustain the 15-Mbytes/sec average transfer rate on all 160 ports simultaneously, giving it an aggregate peak throughput of 2400 Mbytes/sec. In order to achieve this peak rate, the devices on all 160 ports must be sending a steady stream of data to one of the other ports, with a different receiving port for each of the sending ports. To put this in perspective, the X-Bus of the current DSAP has a total bandwidth of 40 Mbytes/sec.

A primary design consideration for the EDSAP has been the ability to interface easily with commercial components and other subsystems. This ability is becoming increasingly crucial in modern processor design. The final report of a recent Lincoln Laboratory-wide study [3] discusses the attributes desirable in multiprocessor radar signal processing architectures:

> *To support the development of experimental and demonstration radars for ground or airborne applications, a simpler set of interprocessor communication standards is both appropriate and desirable. The goals of an interprocessor communication or interface standard for prototype radar applications should*
>
> 1. *Accommodate growth in number and type of processing and I/O nodes (scalable)*
> 2. *Provide bandwidth commensurate with data sampling rates*

3. *Support heterogeneous processing nodes including*
   (a) *Hardwired or partially programmable systolic processors*
   (b) *Programmable DSP, CISC, and RISC processors*
   (c) *I/O devices, such as A/D and D/A converters*
4. *Provide for real-time control and context-switching*
5. *Allow evolutionary, incremental improvements to keep pace with technology*

The report later lists the architectural features that would allow a processor to meet the criteria required to support prototype radars:

Based on the preceding discussion of desirable features for an interface standard, the following architectural constraints have been identified as compatible with the goal of developing an interprocessor communication and interface standard for radar signal processing.

1. *MIMD, private-memory processing nodes. The MIMD constraint ensures that heterogeneous nodal processors can be accommodated. The private-memory constraint avoids the scaling limitations associated with shared memory.*
2. *Industry standards for low-speed data, control, and diagnostics: Industry standards such as the VME bus and its successors should be used whenever possible in order to exploit chip and board-level hardware and software support.*
3. *Bit-serial interconnect for high-speed data. A crossbar switch accommodates scalable, arbitrary interconnect of processing nodes. Subsets of crossbar interconnect include nearest neighbor, mesh, and lattice interconnect. The primary drawback to a crossbar switch is the growth in connections with the number of processing nodes. To minimize the proliferation of interconnections, bit-serial data paths are employed, and the maximum number of data paths per crossbar is constrained. Bit-serial interconnect strategies are compatible with the expected introduction of fiber and free space optical interconnections.*
4. *Programmable interface chip. A major hurdle in developing a robust interface standard is the profusion of specialized interfaces associated with commercially available processor chips and systems. To deal with the range of formats and protocols, a programmable interface processing chip is envisioned to facilitate interfacing the bit-serial interconnect to the arbitrary node processors.*

The EDSAP design embodies all four of these architectural features. The first three were addressed previously. With regard to the fourth feature, one of the PE's processors functions as a programmable interface (i.e., the PE chip can function as a programmable interface chip).

Two application-specific integrated circuits (ASICs) are being designed for the processor. One is the PE chip. In the EDSAP, a "processing element" consists of the PE chip supplemented by some amount of external data memory. In the baseline system, each PE will contain one Megaword (48-bit words) of exter-

nal high-speed static data memory (SRAM). A conservative estimate of the instruction rate for the baseline system is 33 MHz. This corresponds to a 100 MOPS peak capability per PE (each PE can perform one multiply and two ALU operations per instruction). At this rate, a PE can compute a 1024-point complex Fast Fourier Transform (FFT) in 500 microseconds.

The second ASIC being designed is the network switch (NS). Each NS chip is a full crossbar switch for eight bidirectional bit-serial communication channels. A conservative estimate of network and NS chip speed is 200 Mbaud.

The sections of this report that follow provide more detail about the components of the EDSAP: the PEs, the communication network, the VME interface, the diagnostic bus, and the high-speed I/O interfaces. Other topics discussed include (1) a multichip module (MCM) EDSAP system, (2) the size, weight, and power requirements of the baseline and MCM systems, and (3) a survey of candidate applications for the EDSAP architecture.

Appendices A and B provide examples of how the EDSAP architecture could be used in specific systems. Appendix A describes the mapping of SAR processing to the EDSAP architecture. Appendix B describes the implementation of phased array radar processing with adaptive beamforming.

This report is being written before the detailed design of the hardware is complete. It is likely that there will be some changes as the design process proceeds. Also, more detailed documentation is being prepared that will provide the information necessary to use the EDSAP.

2. PROCESSING ELEMENT

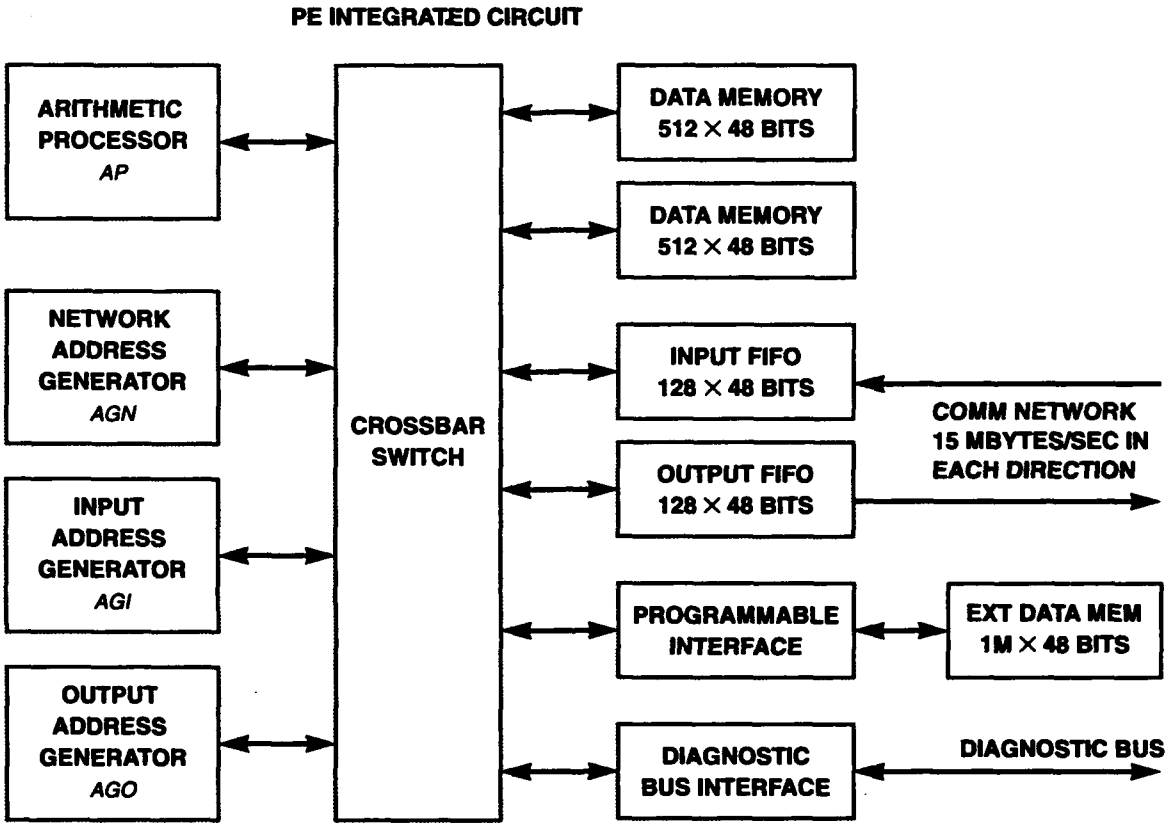

Figure 2-1. Block diagram of baseline processing element.

Figure 2-1 shows the processing element. As mentioned previously, a processing element consists of the PE chip supplemented with external data memory (one million 48-bit words in the baseline system). The PE chip contains one kiloword of internal data memory and four programmable processors: one arithmetic processor (AP) and three address generators (AGs). Each processor contains its own dedicated instruction memory and registers. The PE chip performs the functions of the original DSAP's AP chip, three AG chips, and gate arrays.

The PE's four processors are connected to the three data memories (and to each other) via a crossbar switch. There is a prioritized arbitration scheme for handling conflicts.

The network address generator (AGN) handles the interface between the PE and the communication network. It contains software in ROM used to boot the PE and deal with exception handling. It also contains a collection of system routines that provide the application programs on the AGN with an abstract interface to the communication network. The remaining three processors on the PE—the input address generator (AGI), the output address generator (AGO), and the arithmetic processor (AP)—work in concert to perform the signal processing computations.

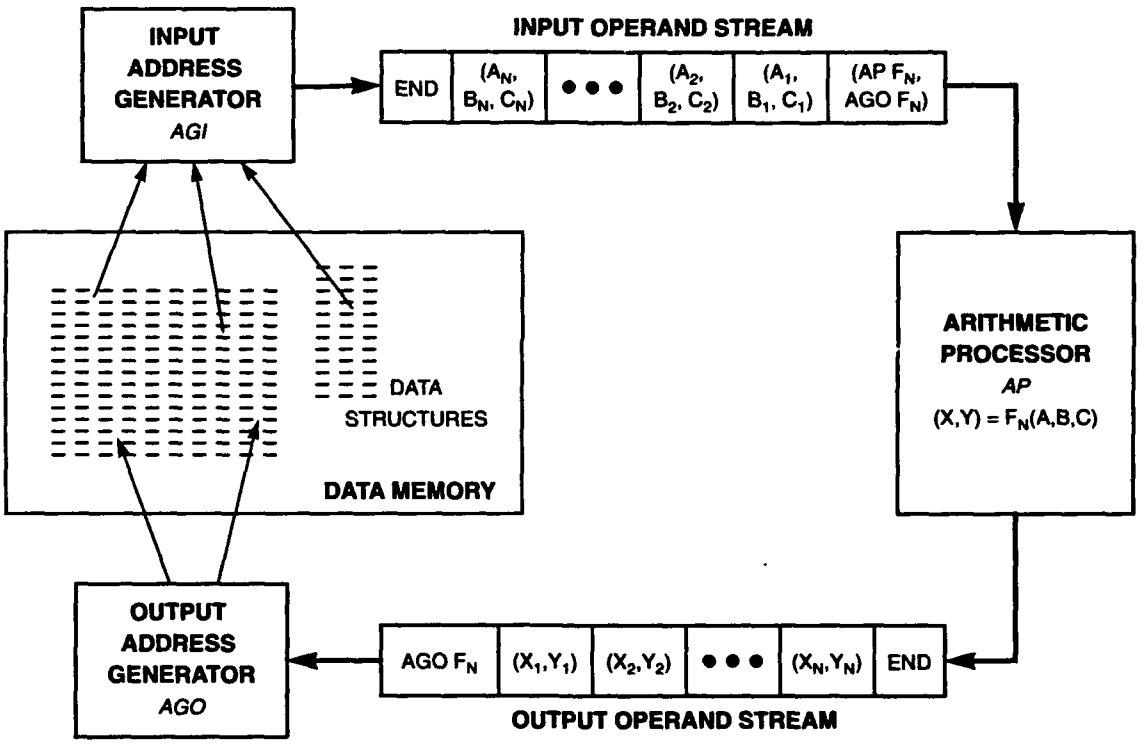

Figure 2-2. Vector (stream) processing.

Figure 2-2 illustrates the data-stream operation, a significant feature of both the DSAP and the EDSAP architectures. The AP performs signal processing calculations without regard to the data memory addresses of the inputs and outputs. The AGI fetches the input stream of AP operands from data memory

while the AGO shepherds the output stream. FIFO buffers on the input and output sides of the AP provide rate buffering; thus, each address generator need only match the average data rate of the AP to keep the AP running at full efficiency. This separation of the address calculations and arithmetic calculations into asynchronous processes (via the FIFOs) simplifies the optimization of complex signal processing algorithms.

The AP can perform three operations per instruction: one multiply and two ALU (add, subtract, compare, or logical). This capability provides the PE with a 100-MOPS peak processing capability at the expected processing rate of 33 MIPS. The ratio of two adds to one multiply yields a very efficient implementation of the radix-4 FFT, which is used in both the DSAP and the EDSAP. The 100-MOPS PE will be able to perform a 1024-point complex FFT in 500 microseconds. Logic in the ALUs provides for efficient thresholding (compare) operations. The EDSAP PE will perform 24-bit fixed-point arithmetic. For a typical signal processing operation, the 48-bit data memory word will contain a complex number with 24-bit real and imaginary parts.

As mentioned previously, the PE chip can be used to interface commercial and other devices to the EDSAP's high-speed communication bus. For example, the PE chip can be interfaced to another processor through a memory shared by the PE and the other processor.

2.1 COMPARISON OF EDSAP PE WITH DSAP PE

The EDSAP PE will contain three data memories, one external and two internal, as opposed to the DSAP's single data memory. The DSAP PE's data memory is accessed twice per instruction cycle; by contrast, each of the EDSAP PE's internal data memories will be accessible once per instruction cycle. The external data memory will require at least one cycle to access, but may require more than one cycle depending on the relative speeds of the memory ICs and the PE chip. Plans for the baseline system call for memory fast enough for the PE chip to access once each cycle.

The PE chip will be able to address up to 15 million 48-bit words (90 Mbytes) of external data memory; a one-million-word external memory is planned for the baseline PE. This external memory can be either static or dynamic RAM. External RAM access timing will be programmable. There will also be programmable configuration registers for setting up the logic for refreshing dynamic RAM. By contrast, the DSAP PE has a total of 64K 32-bit words of static data memory.

The AGX in the DSAP PE controls the PE's access to the X-Bus, a high-speed parallel bus used to move data into and out of the PE. In the EDSAP PE, the AGX will be replaced by the AGN, and the X-Bus will be replaced by the high-speed serial network (see Figure 2-1).

The AP currently in use in the DSAP can simultaneously receive an input and generate an output to data memory each cycle. The DSAP PE does not take advantage of this capability, however, and therefore can only perform a single AP input or output each cycle. In the EDSAP, the PE will support both an AP input and output during each instruction cycle.

3. COMMUNICATION NETWORK

As mentioned previously, PEs will be connected to each other as well as to external I/O devices via a number of high-speed bidirectional bit-serial communication channels. The channels will be interconnected via a number of network-switch (NS) ASICs. The channels and the network switches will constitute the communication network. Each network switch will be able to connect up to eight bidirectional channels via its internal crossbar switch. The ratio of PEs and external I/O channels to network switches will determine how closely a communication network implementation approximates a full crossbar switch.

As an example, consider a system consisting of 32 PEs and 16 bidirectional fiber-optic I/O lines. Assume that four PEs and one NS reside on each of eight processor cards, which plug into a backplane. There is a total of 48 communication ports divided into two groups: 32 reside on the processor cards and 16 reside on the backplane. The 16 ports on the backplane interface the system to external I/O via fiber-optic translators. Figures 3-1 and 3-2 present two possible 48-port network topologies. The low-cost topology in Figure 3-1 imposes performance limitations and provides relatively little redundancy; the topology

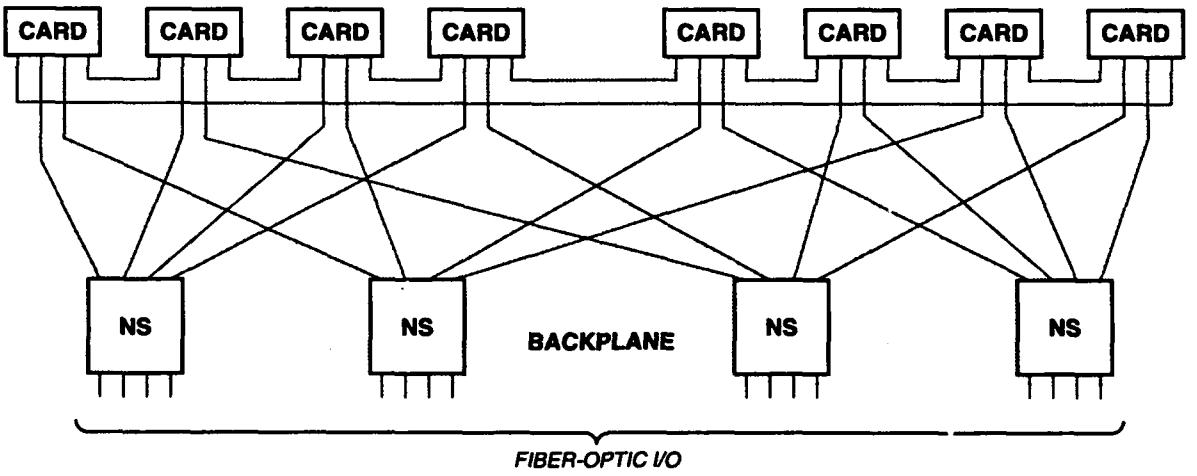

Figure 3-1. Topology of low-cost communication network.

in Figure 3-2 provides high performance and a high degree of redundancy. Although both networks can support 48 simultaneous 15-Mbytes/sec (720 Mbytes/sec total) bidirectional conversations, their overall capabilities are quite different. The difference between the two networks comes from a difference in the number of NSs residing on the backplane (i.e., external to the processor cards). The low-cost network has four backplane NSs, while the high-performance network has sixteen.

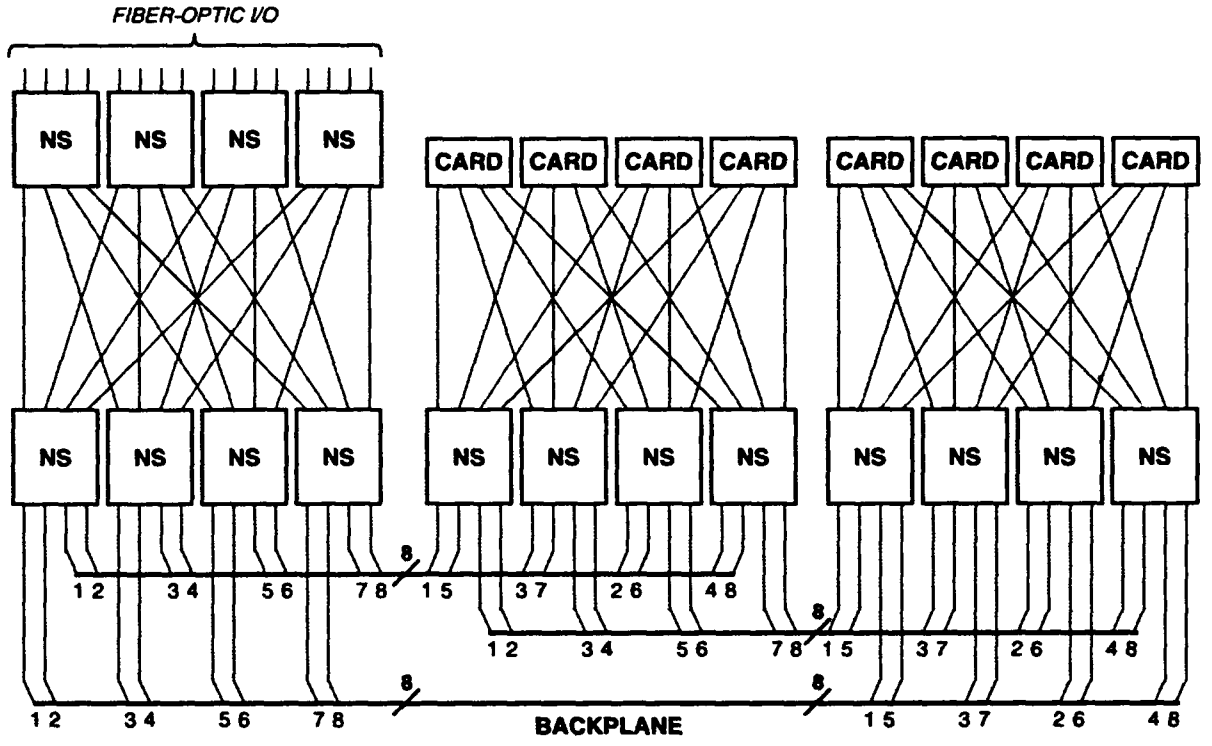

Figure 3-2. Topology of high-performance communication network.

Figure 3-1 depicts the low-cost network topology. Each of the 16 fiber-optic I/O ports connects directly to four processor cards via one of the four backplane NSs. For an I/O port to communicate with any of the remaining four cards, the information must be routed through one or two additional network switches (assuming that the NS on each card can route data between the four ports on the card). It is important to realize that, with this topology, if an I/O port must communicate with a card it is not directly connected to, the resources available to other ports on the network will be reduced.

In Figure 3-1, each card is connected to its two nearest neighbors. If cards transmit only to their nearest neighbors, each card can transmit at the full 15-Mbytes/sec rate without interfering with another card or I/O. If cards transmit to other than their nearest neighbors, however, they reduce the network bandwidth available to other ports.

Figure 3-2 shows the higher-performance network topology. This network can be envisioned as three interconnected subnets. Each subnet implements a full crossbar; within each subnet, all 16 ports can send to any of the other ports simultaneously without interfering with the others. The left-most subnet han-

dles the I/O ports. The full I/O port crossbar is implemented to support interconnection of multiple chassis. The other two crossbar subnets interconnect the processor cards.

Each of the three subnets is connected to the other two via eight serial channels. The subnets can communicate at 120 Mbytes/sec without interference as long as two or more originating ports do not attempt to simultaneously transmit to the same destination port.

The current plan is to interconnect the 160 ports of the baseline system (Figure 1-1) with a full-crossbar network similar to that in Figure 3-2. As applications are explored in greater detail, the actual number of PEs and I/O ports may change.

3.1 PACKET SWITCHING

The network can be thought of as a store-and-forward packet-switched network. In addition to the actual data, each packet will include a description of the packet type and a destination address, as well as information to identify the packet and minimize the probability of transmission errors. Any packet not destined for a nearest neighbor will be routed from point to point until it reaches its final destination. Any node (PEs and NSs are nodes) in the network can be the final destination of a packet; intermediate points are always NSs. Packets can vary in size from 18 to 773 bytes.

The path that a packet follows through the network will be determined by a series of addresses within the packet's address section. When the packet arrives at a network switch, the switch will read the current address, indicated by a pointer field contained within the address section. If the NS is not the packet's final destination, the NS will send the packet out the port indicated by the current address and increment the pointer; if the NS is the final destination, it will execute any requests (e.g., data reads/writes, status inquiries) or commands contained in the packet. These requests/commands will be contained in a separate field. All the requests/commands applicable to the NS will be defined in hardware. In the PE, some of the requests/commands will be defined in hardware; the rest will be definable by applications software.

The above procedure will send a packet from a single origin to a single destination. When a packet is to be distributed to multiple destinations, its address section will contain only a route number, and a "route" bit in the packet's header will be set. When an NS receives such a packet, it will refer to a lookup table stored in its routing address RAM to determine which port(s) should output a copy.

3.2 NETWORK SWITCH

Figure 3-3 is a simplified block diagram of a network switch. A serial input stream received at a data port will be converted to 8-bit parallel bytes, buffered in a 2-Kbyte FIFO, routed through an 8×8 crossbar switch to the specified output port, converted back from parallel to serial form, and then transmitted. (Refer to Figure 3-4 for a simplified diagram of a port.) The 8×8 crossbar switch will allow packets received at any of the eight data ports to be sent out any of the eight ports. The network switch will handle data entering all eight ports simultaneously unless a desired output port is busy; packets destined for the busy port will wait in FIFOs. An external, electrically erasable, programmable read-only memory

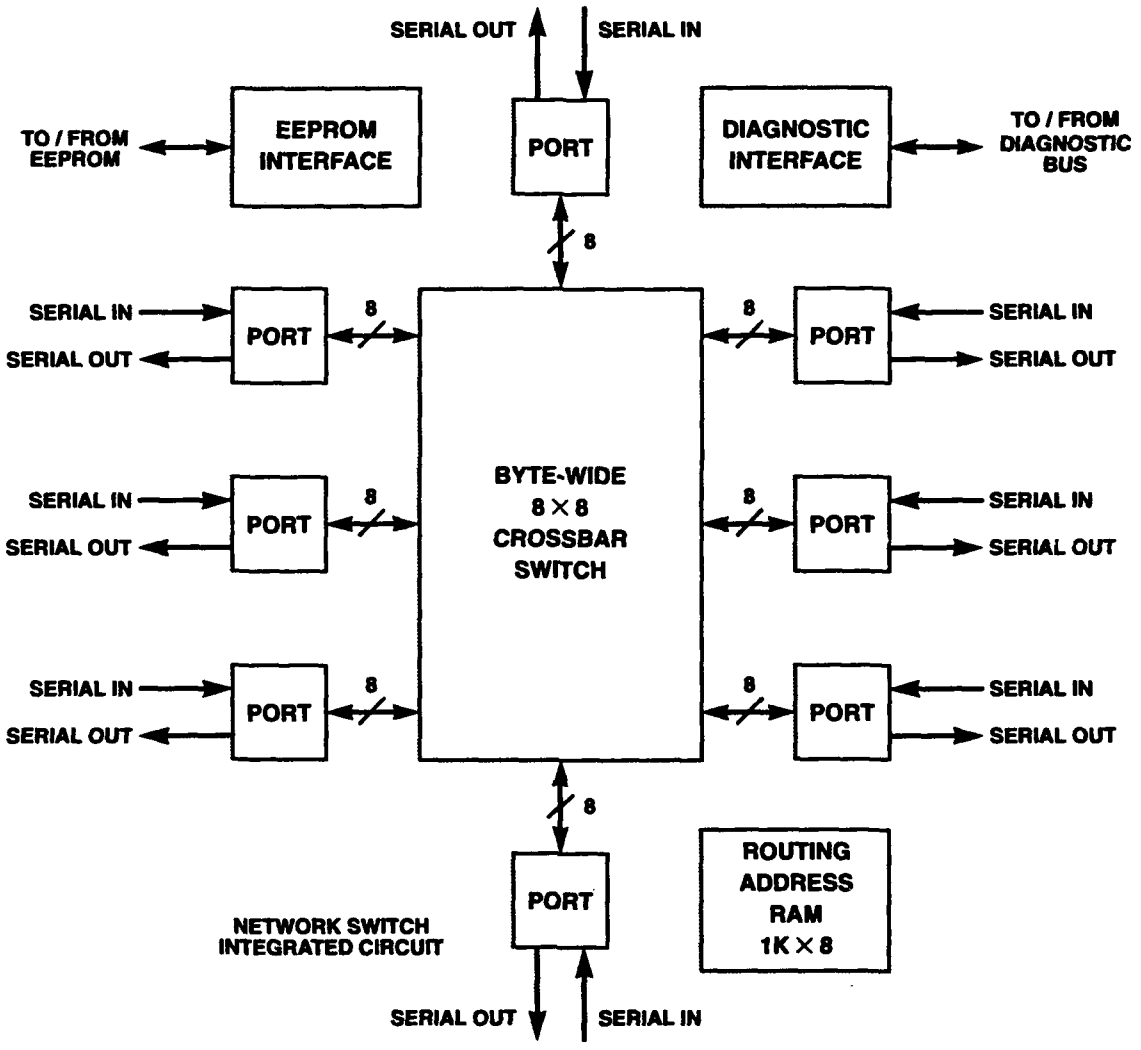

Figure 3-3. Simplified block diagram of network switch.

(EEPROM) will store configuration data for NSs and PEs. For testing purposes, the FIFOs of all eight ports as well as the route address RAM can be accessed via a diagnostic bus.

Control logic not shown in the figure will determine what to do with each packet and then do it; this will also occur when errors are encountered in the packet. The NS will log these errors and other operational data. Registers accessible to the network, and thus to the host, will allow each port to be configured individually to various performance and protocol requirements.

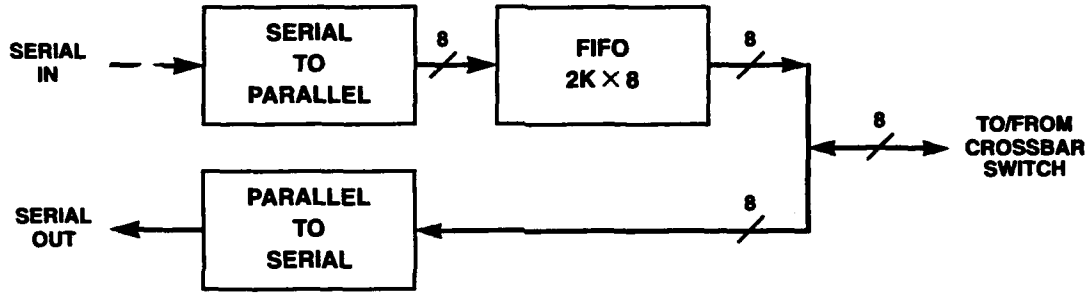

Figure 3-4. Simplified block diagram of a network switch port.

4. PROCESSING ELEMENT BOARDS

High-level design of two types of PE boards is in progress. One design contains six PEs and the other 16 PEs. Each board will be the size of a 6U VME board (6.3 in. × 9.2 in.). The hex PE board (six PEs) is planned for the baseline system. The 16-PE board will require the use of multichip module (MCM) technology.

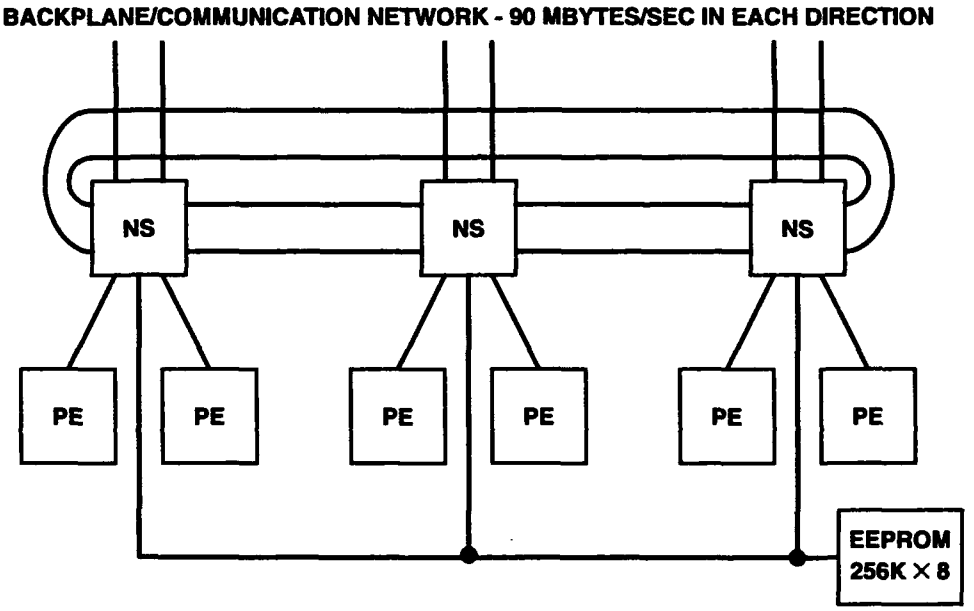

Figure 4-1. Block diagram of hex PE board.

4.1 HEX PE BOARD

Figure 4-1 is a block diagram of the hex PE board. The board's six PEs will provide a total of 600 MOPS. The PEs will communicate with each other and with the rest of the system through the three network switches. There will be enough communication bandwidth for all six PEs to be sending data at 15 Mbytes/sec to any one of the other PEs simultaneously (assuming that each PE has only one other PE sending to it at a given time). Alternatively, there is enough bandwidth for all six PEs to be sending data off the board and for all six to be receiving data from off the board simultaneously (i.e., 90 Mbytes/sec in each

direction). In later systems with more PEs per board, it is likely that there will be fewer ports between the board and the backplane than there are PEs.

The EEPROM connected to the NSs will store configuration information. At power-up, the AGN on each of the PEs will prompt the network switch to transmit the contents of the appropriate locations within the EEPROM. Configuration information will include how much external data memory is present for each PE, as well as the timing necessary to access the memory. The EEPROM will also be able to store programs. The EEPROM will be programmable by sending appropriate commands to one of the NSs.

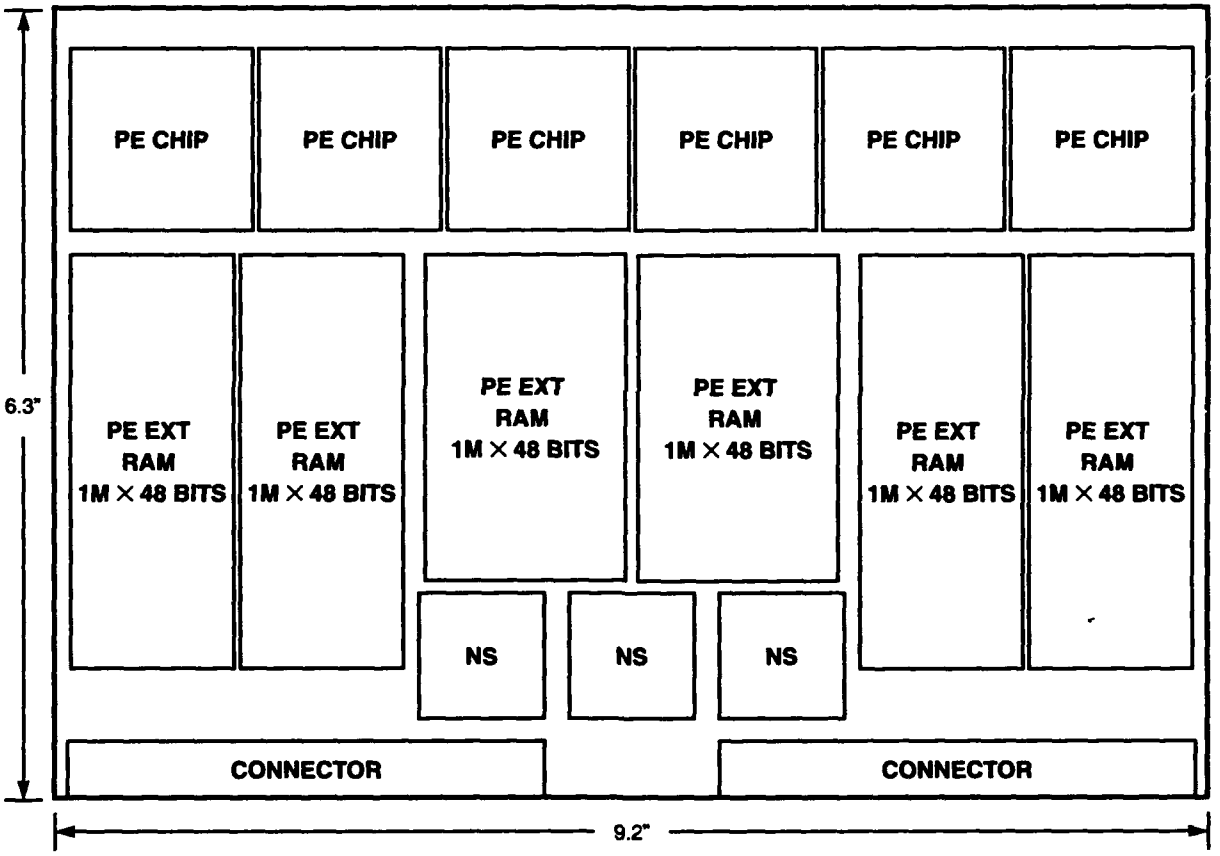

Figure 4-2. Floor plan of hex PE board.

Figure 4-2 shows the preliminary floor plan of a hex PE board. Different PE boards might be designed for different applications. Memory-intensive applications might have fewer PEs per board but more data memory per PE. Applications requiring floating-point calculations might have a commercial DSP, RISC, or CISC processor as a coprocessor for each PE. Because the PE and NS will contain most of the logic, the design of PE boards is relatively straightforward.

In the floor plan shown, the PE chips are in pin-grid array sockets (on only one side of the board). Each PE's external RAM consists of twelve 4-Mbit SRAMs that are in SOJ (small outline packages with J leads) packages on both sides of the board. The NSs are in sockets on the same side of the board as the PE chips. The blank area near the two connectors is used for approximately 10 to 15 other ICs including the EEPROM; they are surface-mounted on both sides of the board.

4.2 16-PROCESSING-ELEMENT BOARD

A PE board using MCM technology has been sized. By using conservative MCM technology, it should be possible to fit a PE with its 1M × 48 bits of external memory into an MCM approximately 2.5 in. × 2.75 in. Sixteen of these MCMs with supporting circuitry should fit on a board the same size as the hex PE. When techniques to stack three or four memory dies on an MCM have matured, it is likely that 32 PEs with their memories and supporting circuitry can fit on the same size board as the hex PE.

5. VME INTERFACE

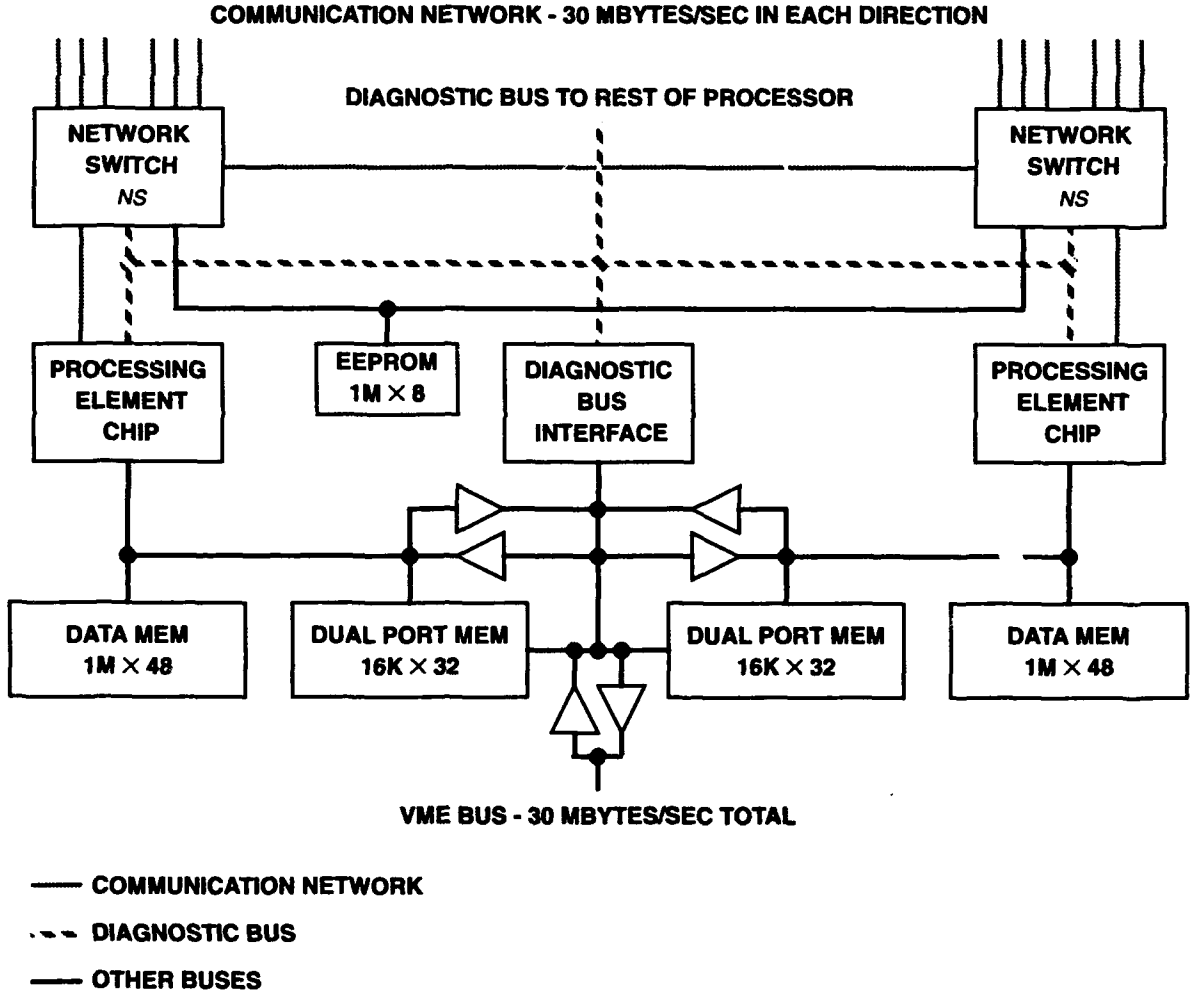

Figure 5-1. Block diagram of VME interface.

The VME interface (VMEINT) will be a VME board with two PEs and two NSs, as shown in Figure 5-1. The VME interface will connect to the rest of the processor via the communication network. It can be thought of as an interface between the EDSAP's serial communication network and the VME bus.

It will be possible to have more than one VME interface in a system. In fact, the EDSAP's serial communication network may be used to interconnect a large number of VME chassis.

The VME bus has been chosen for the baseline system because many commercial products use it. If it becomes desirable to interface to a different bus, it should not be difficult to modify the interface design.

The VMEINT will include two dual-port memories; one port will be used by one of the PEs and the other port will be used by the VME bus. Each of these memories will be a true dual-port memory; two accesses may occur simultaneously as long as both ports are not used to access the same location at the same time. The timing of each port will be asynchronous with respect to the other.

The VMEINT will include hardware to support semaphores that can be used for arbitrating whether the VME bus or the PE is allowed to use a particular block of memory. There also will be hardware to allow a processor on the VME bus to interrupt the AGs within either of the PEs and to allow either of the PEs to cause VME interrupts.

Each of the PEs will be able to make one access every two cycles (i.e., 16.5 MHz assuming a 33-MHz instruction rate) to its external RAM or its dual-port RAM. The VME bus will be able to make block transfer accesses to either of the dual-port RAMs at 7 to 8 MHz (32 bit words), which approaches the limits of the VME bus; thus, the total bandwidth between the VME bus and the dual-port memories will be approximately 30 Mbytes/sec. Each of the PEs will be capable of transferring data between the communication network and its dual-port RAM at 15 Mbytes/sec in both directions simultaneously. Thus, there will be a total bandwidth of 30 MBytes/sec between the VME bus and the rest of the EDSAP with the VME bus block transfer rate being the limiting item. (The VME interface for the DSAP was capable of only about 3 MBytes/sec.)

The VME interface will support both 2-byte and 4-byte accesses but not single-byte accesses. Accesses must be aligned on 2- and 4-byte boundaries, respectively. The initial version of the VMEINT is expected to be a slave on the VME bus, assuming that some other processor board on the VME bus will be the master.

The two PEs on the interface board will be capable of delivering 200 MOPS of processing power. For some applications this may be sufficient to perform all the signal processing. In fact, the MTI radar system successfully demonstrated at Fort Sill, Oklahoma, in the spring of 1990—which was capable of only 108 MOPS—was able to process moving-target detection on a 10-km swath (250 40-m range cells) at a pulse repetition frequency (PRF) of 6250 Hz (1,562,500 samples per second) [2]. The VMEINT of the EDSAP, along with a radar interface (see Section 6), could replace that DSAP VME interface, CE (control element), and six dual-PE cards.

5.1 DIAGNOSTICS AND SOFTWARE/HARDWARE DEBUGGING

The VMEINT will interface the VME bus to the EDSAP's diagnostic bus. A processor on the VME bus will be able to use the diagnostic bus to access the external memory (including EEPROM), internal RAM, and registers of all PEs and NSs within the EDSAP. A processor on the VME bus will also be able

to access RAM and registers within each of the PEs on the VMEINT directly (i.e., without using the diagnostic bus). In the past such features have proven invaluable for diagnosing chip fabrication problems.

The diagnostic bus will be bit-serial and will go throughout the EDSAP. It will be based on the Extended Serial Digital Signal (ESDS) Testability Bus Subset of IEEE Standard *P1149 Testability Bus Specification* [4]. The diagnostic bus is intended for the following purposes:

1. testing the NS at time of fabrication (most of the PE chip testing will be performed using its parallel port);
2. loading and running diagnostics for testing subsystems or the entire system;
3. loading the EEPROM associated with the NSs;
4. loading and debugging software;
5. disabling components that have been found to be faulty.

5.2 CLOCK GENERATION

Two oscillators planned for the VME interface will generate clocks for the entire EDSAP. One clock will define the bit rate over the serial network, and the other will define the instruction rate of the PEs. If space permits, the VMEINT will include two frequency synthesizers for generating these clocks. These two frequency synthesizers would be under program control of the VME bus-based host. The frequency of operation would be varied while diagnostics are run in order to verify margin and expose any problems that only occur at a particular frequency range. (Testing the DSAP while running its clock from an external sweep generator proved invaluable in bringing problems to light and helped make the DSAP very reliable.)

6. RADAR INTERFACE

An interface to a specific radar is yet to be designed. Figure 6-1 shows a possible interface. For this example it was assumed that the radar has multiple A/D channels with bursts of data at 200 MHz, and an average data rate of less than 60 Mbytes per second per A/D channel (this is consistent with a synthetic aperture radar). The example also includes interfaces to radar control logic and a source of navigation information.

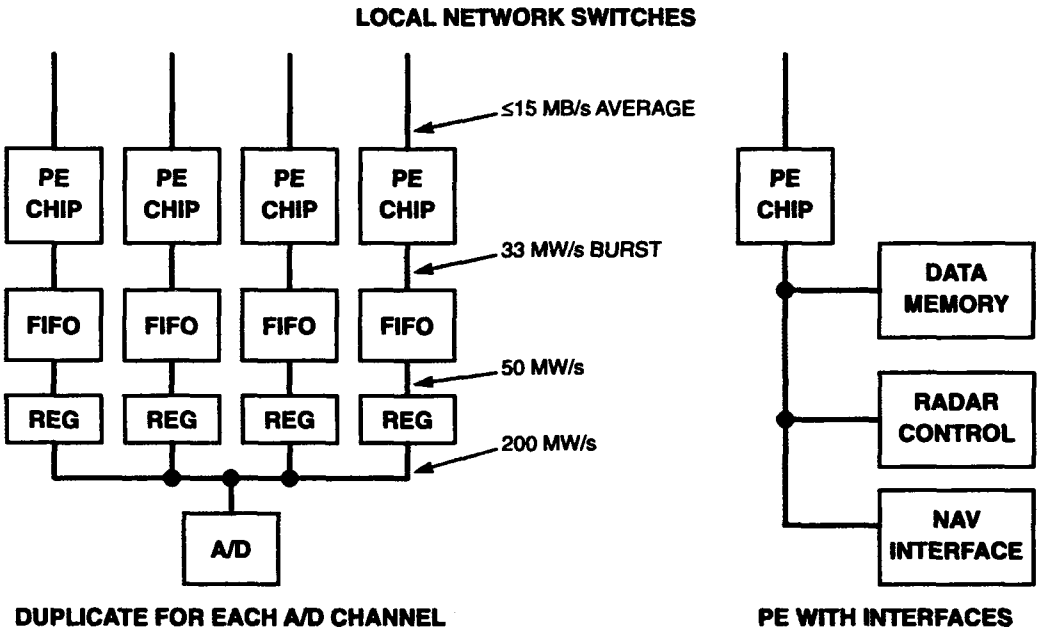

Figure 6-1. Example of a radar interface.

6.1 PATH OF VIDEO

The video from each channel of the radar is digitized by an A/D converter. The A/D converter outputs data at a continuous rate of 200 MW/s (million words per second). This 200-MW/s stream of data is sent to a set of four registers. The registers are clocked round-robin, creating four streams of data at 50 MW/s each. These 50-MW/s streams are sent to synchronous FIFOs whose inputs are gated to select the desired range gates, creating 50 MW/s bursts of data into the FIFOs. The FIFOs reduce the burst data rate

to 33 MW/s, a rate that the PEs can handle. The PEs then format the data streams, add any control information (e.g., time tags), and send them to the main part of the processor. The PEs could possibly perform some very simple filtering before they send the data to the main part of the processor. When used for interfacing, the PE chip does not require any external memory.

A/D burst rates higher than 200 MW/s can be processed by using either faster FIFOs or additional register/FIFO/PE groups. If the average rate exceeds that of four PEs (approximately 15 Mbytes/s per PE), additional register/FIFO/PE groups can be added. If the average data rate is low, more than one register/-FIFO pair can be connected to each PE.

Each PE has a programmable 48-bit parallel port. If the average data rate out of the A/D is 15 Mbytes/s or less, all four register/FIFO pairs can be connected to one PE (as long as the number of register/-FIFO pairs $\times$ the number of bits output by each pair ≤ 48).

At even lower data rates, further simplifications are possible. For example, the burst data rate in the UAV radar's moving target indicator (MTI) application was approximately 15 MW/s (two 8-bit samples per word, using two A/Ds) and the average data rate was approximately 1.6 MW/s (3.2 Mbytes/s). At this low data rate both A/Ds could be handled using only a single PE chip.

6.2 RADAR CONTROL AND NAVIGATION INFORMATION

In addition to the PEs used for interfacing with the A/Ds, the subsystem shown in Figure 6-1 contains a PE for interfacing with radar control logic and navigation instruments. The task of generating detailed timing signals for the radar could be accomplished in several ways. At one extreme, the PE would generate all the timing signals by counting cycles. The PE would count cycles and change bits in a register; these bits would constitute the various timing signals. The other extreme would have external counters generate all the timing signals; in this case the PE would simply write registers controlling the counters. Most systems would use a combination of these techniques.

PEs will be configurable so that one or more of their AGs can be interrupted in response to external signals. This feature will make it possible to synchronize the PEs in the radar interface with each other and with external circuitry.

In order for an airborne radar to function properly it needs navigation information. The radar interface shown in this example includes a navigation interface that receives information about the location of the platform. This information would then be associated with the corresponding A/D data, and both the A/D and navigation data would be sent to their destinations on the communication network.

7. MODULAR DESIGN OF EDSAP

The design of the EDSAP is modular at the chip, board, and chassis levels. The intent is that the modules can be configured differently to meet the needs of many systems. The designs of the ICs and boards will be available in a computer aided engineering (CAE) environment. A designer using a compatible CAE system will be able to make changes to the modules.

7.1 CUSTOM ICs

The ICs are being designed using a silicon compiler from Cascade Design Automation on a Mentor Graphics workstation. There is a high non-recurring engineering (NRE) cost associated with modifying these ICs; changes should be avoided if practical. Two ASICs will be developed for the baseline EDSAP:

1. the processing element (PE) will contain three AGs, an AP, internal memory, and an interface to the communication network. It will be capable of 100 MOPS. In addition to being the computation engine for the EDSAP, it will be able to interface devices to the communication network. For example, it will be able to interface to microprocessors, floating-point DSP chips, and special- purpose filter chips;
2. the network switch (NS) will be used to implement the EDSAP's store-and-forward, packet-switched communication network. It will contain FIFOs and a eight-port crossbar switch and will interconnect eight ports on the communication network.

7.2 PRINTED CIRCUIT BOARDS

The printed circuit boards (PCBs) are being designed using a Mentor Graphics workstation. Four PCB designs are planned for the baseline EDSAP:

1. The VMEINT PCB will interface the EDSAP to the VME bus. It should not be difficult to adapt this design to other buses;
2. the hex PE PCB will contain six PEs, each with $1M \times 48$ bits of external data memory. The board will also include three NSs. It should be fairly straightforward to change the amount of data memory used by any of the PEs. It will also be possible to add coprocessors to any of PEs;
3. the backplane PCB will be the EDSAP's backplane. It will contain the NSs necessary to interconnect the other boards and external interfaces;
4. the radar interface PCB—its performance and form factor will depend on the target system for the baseline processor.

8. SIZE, WEIGHT, AND POWER

Two versions of the EDSAP have been sized. The baseline version is based on a single chassis with 21 hex-PE boards. The multichip module (MCM) version is based on two chassis, each with 21 sixteen-PE boards. For each version, a separate six-slot VME chassis is included for the host.

TABLE 8-1
EDSAP Size, Weight, Power, and Other Parameters

Parameter	Baseline version	MCM version
Number of signal processing chassis	1	2
Volume (ft^3)	3	5
Weight (lb)	70	150
Power (W)	1,000	4,000
Number of PEs	126	672
Throughput in GOPS*	12	67
Data memory (Mbytes)	800	4,000

* Billions of operations per second

The chassis and backplanes for both versions of the EDSAP are expected to be the same. The PE boards are the same size as 6U VME boards (6.3 in. × 9.2 in.), so substantial amounts of standard VME hardware are expected to be used. The backplane will be a custom design and will include approximately 91 NSs, forming the communication network among PE boards. Each chassis will include power supplies capable of providing approximately 2.5 kW. The 19-in. rack-mount chassis is expected to be approximately 15.75 in. high, 17.5 in. wide, and 12.75 in. deep. The extra 5.25 in. in height beyond the typical VME chassis is to duct cooling air. The chassis can be redesigned to fit the form factor of target systems.

The power and performance estimates given here are based on the PE and NS being fabricated in a 0.8-μm, 5-V CMOS process. If industry experience with 3-V CMOS processes advances rapidly enough, the PE and NS will be fabricated in a finer-geometry 3-V CMOS process. Because a silicon compiler is being used to design the chips, processes can be changed fairly easily. When the chips are ready for release to fabrication, they can be recompiled in the best process available.

9. SOFTWARE DEVELOPMENT ENVIRONMENT

The support software for the EDSAP is expected to build upon that of the DSAP. The DSAP software development environment is provided by three programs: a functional simulation of the hardware [5], a debugger [6], and an assembler for writing programs [7].

Source programs for the application software are written in C. These programs are translated into object code by the assembler, which consists of a collection of functions that fill the microcode fields of the target processor. The debugger accepts the object code as input and serves as the user's interface to either the actual hardware or the software simulation. Typically, the debugger is used both for testing and debugging programs on the simulated processor and for loading and executing them on the hardware.

The simulated processor provides a functional register-level abstraction of the PEs (but not their interaction via the DSAP's high-speed data bus). It provides a complete programming model of the hardware, replicating the functionality of the hardware on an instruction-cycle by instruction-cycle basis. Because much of the detail irrelevant to programming has been discarded, the simulation executes on an engineering workstation quickly enough to be useful to an application developer. Also, the simulation allows the developer to examine states within the processor that are not readily accessible in the hardware.

The functional simulation of the DSAP was developed independently of the CAD hardware simulation used to develop the hardware. Therefore, verifying the correctness of the simulation required a great deal of painstaking effort. Improvements in workstations and CAD tools should allow the development of a functional simulation of the EDSAP PE that is derived directly from the hardware simulation.

10. CANDIDATE APPLICATIONS

While the EDSAP is specifically targeted to radar applications, it is also being designed to be applicable to a wide variety of other applications. The remainder of this section outlines several potential radar and non-radar applications.

10.1 HIGH-RESOLUTION SAR (SYNTHETIC APERTURE RADAR)

The EDSAP would be a very good fit for high-resolution SAR applications. Appendix A gives a detailed system sizing for a SAR system that includes automatic target recognition.

10.2 PHASED ARRAY RADAR WITH ADAPTIVE NULLING RADAR

The EDSAP would be a good match to applications requiring a large number of input channels, such as phased array radar. Appendix B gives detailed examples of the sizing of EDSAP-based systems for 8- and 64-channel phased array radars with adaptive nulling.

10.3 SUPERCOMPUTER

The EDSAP's network could be used to connect a large number of processors to form a supercomputer. The EDSAP PE chip would be used to interface the individual processors to the communication network and hence to the rest of the system. The processors within the PE chip could deal with much of the overhead associated with communicating with other processors in the system. Such a system could take many forms. For instance:

1. Each "processor" might actually be a small number of processors on a VME bus. In this case an EDSAP VME interface board would be used to interface each VME bus to the rest of the system. In this way a large number of these VME systems could be connected to form a large system.
2. An interface board based on a PE chip could be designed for the internal bus of the workstations, to interface them to the EDSAP's communication network. Using the EDSAP's communication network to tie workstations together would provide much higher performance than that of local area networks currently in use.
3. A custom module could be built that contains a commercial reduced-instruction-set computer (RISC) processor interfaced to the EDSAP's communication network via a PE. A large number of these modules could then be interconnected.

Any of the three systems described above might use RISC processors as the individual processors. One way to interface a PE with a RISC processor would be to give the PE access to the RISC processor's memory. This would allow a shared-memory scheme in which a number of RISC processors used the EDSAP's communication network to access the memory of all the other processors. This scheme could be implemented by running a virtual memory system on each RISC processor. The RISC processor would know which pages were in its local memory. If an application program tried to access a page of memory

that was not in the local RISC processor's memory, the local PE would get the desired page, put it in the local RISC processor's memory, and let the RISC processor know that the page was there.

10.4 LOCAL AREA NETWORK HUB OR BACKBONE

The NS will be designed to support Ethernet, MIL-STD-1553B, and FDDI communication standards. In one configuration, the NS would break up the bit stream from a foreign network into packets sized appropriately for an EDSAP, then attach the header and trailer information necessary for sending this packet to an EDSAP PE. All the traffic from a particular foreign network would be sent to one PE, which would decide where to send it.

Foreign networks not supported by the NS could be interfaced to an EDSAP-based network by using a PE along with an interface module intended for the foreign network.

10.5 TELECOMMUNICATIONS SWITCH

An EDSAP network could be us 'o interconnect a large number of telecommunications channels. The communication channels to be int. nnected would be interfaced to an EDSAP communication network using PE chips. The EDSAP communication network would then be used to route traffic among the channels.

10.6 HDTV (HIGH-DEFINITION TELEVISION) STUDIO SWITCH

The 15-Mbyte/sec data rate of an EDSAP port should be adequate to carry a digital HDTV video signal. A PE chip would be used to interface each of the video sources to an EDSAP port. The EDSAP network would then route the video among ports. Also, the signal processing capability of the EDSAP could be used to generate special effects.

APPENDIX A
EDSAP SIZING FOR SAR

A.1 INTRODUCTION AND SUMMARY

This appendix describes an image-formation and target-detection processor for the Lincoln Laboratory 35-GHz SAR using the EDSAP architecture. It will be assumed that the reader is familiar with the material presented in the Executive Summary and Introduction of this report. The key upgrades from the DSAP to the EDSAP architecture will be mentioned, followed by a discussion of the processor. The image-formation process includes an autofocus step; some assumptions are made concerning the form that autofocus may take. Image formation will produce three images (e.g., HH, VV, and VH), which can be used for detection and/or combined to form a polarimetric whitening filter (PWF) image.

The EDSAP architecture represents a significant improvement over the DSAP. While the individual processing element (PE) is quite similar to the DSAP's PE, the I/O and the interconnections among the PEs have been substantially enhanced in the EDSAP architecture. While all of the DSAP PEs were connected to a single high-speed data bus, the PEs and I/O interfaces in the EDSAP are interconnected using a number of 15-Mbytes/sec bidirectional channels linked by custom network switches. This permits a scalable architecture in which the I/O and interprocessor communication bandwidths can grow with the number of PEs.

As mentioned earlier, each EDSAP PE is much like a DSAP PE except that its data memory will contain 16 times as many complex words (one Megaword). Also, the wordlength has been increased from 32 bits to 48 (the PE will perform 24-bit fixed-point arithmetic). The EDSAP PEs are expected to run three times as fast as the 10-MIPS DSAP PEs. Finally, the DSAP PE's AGX (the processor that handles the interface between data memory and the high-speed bus) is being replaced by an AGN, which interfaces the PE to the network. The differences between the DSAP's single-bus architecture and the EDSAP's networked architecture result in a significant change in functionality between the DSAP PE's AGX and the EDSAP's AGN.

The SAR digital processing involves forming three images in realtime. Each image represents a 375-m-wide stripmap with 1-ft resolution. Each pixel in the image represents approximately $\frac{1}{4}$ m $\times$ $\frac{1}{4}$ m sample spacing. The image is fabricated as a "frame": 2048 range samples wide by 512 pulses long (the data have been "resampled" down from 1500 Hz to approximately 500 Hz; the latter represents pulses in this document). The time required to collect a frame of data is between 0.94 and 1.17 sec and is a function of platform speed. The PWF is applied to the (complex) images to form a fourth, reduced-speckle image. This image is fed to the constant false alarm rate detector (CFAR) process, which finds "regions of interest" at a rate of about six per second (this corresponds to a false-alarm rate of 100 per square kilometer). All four images are then available for texture discrimination and target identification.

The processor design uses 37 PEs (see Figure A-1). Five are allocated for range processing, which involves computing (I, Q) samples from real inputs and then computing the range compression FFT. The data are then shipped to one of four PEs to be used for the autofocus function. Each of the four PEs con-

tains one fourth of the range swath. While these PEs will contain data for all three polarizations, only one polarization needs to be processed for autofocus estimates. The data and the autofocus estimates are then transmitted to one of 16 PEs for cross-range processing. Each of these PEs creates three images for one sixteenth of the range swath. The complex pixels from the three images are combined (using the PWF) to form a fourth image with reduced speckle. The four images are then transmitted to one of twelve PEs allocated for detection and identification. The size of the detection and identification section of the processor is driven by memory requirements.

The sizing discussed in this appendix was done very conservatively, in part because the amount of data memory required drives the sizing in most cases. This leaves a good deal of computing power and PE I/O bandwidth in reserve. The range-processing step is computation-bound and was sized to use 80% of the computing power available in the five PEs. The maximum input or output data rate to a single PE is 4.7 Mbytes/sec (the input rate to each autofocus PE). The channel capacity is 15 Mbytes/sec. Figure A-1 shows the processing flow along with the amount of data memory required for each PE. The interconnections among the PEs as depicted in the figure represent a low-risk design with low utilization of the overall EDSAP interconnect capability.

A.2 IMAGE FORMATION

A.2.1 Range Processing

At the start of range processing, data is input from three polarizations (e.g., HH, VV, and HV). For each pulse at each polarization, 4096 11-bit A/D samples are input to an eight tap FIR filter to form 2048 complex (I, Q) samples. The inputs represent samples of an intermediate frequency, taken after the chirp radar has been deramped. They are referred to as range offset video. Practically, the 4096 sample input vector is decimated. Every other sample becomes an in-phase value to be paired with an adjacent input sample as the quadrature component. The eight tap filters are low-pass to attenuate high-frequency artifacts of the process. Also, the in-phase filter coefficients are offset in time relative to the quadrature filter coefficients to time-align each (I, Q) pair. The first part of range processing requires 144 MOPS.

The second step in range processing is the range-compression FFT. In this step the 2048 complex samples from each channel are input to a Taylor-weighted FFT. The outputs of the FFTs (2048 complex range samples for each of the three polarizations) are sent to a collection of PEs that serves as the corner-turn bulk memory and performs the autofocus function.

In this design, a single PE will perform the entire range-processing function for a single pulse. Five PEs will be allocated to the task, with pulses allocated to PEs in round-robin fashion. The input data rate (from three channels) is 18.8 Mbytes/sec (3.77 Mbytes/sec for each of the five PEs). The output data rate is the same. Each of the range-processing PEs partitions its outputs in range, sending one fourth of the range swath to each of the autofocus PEs.

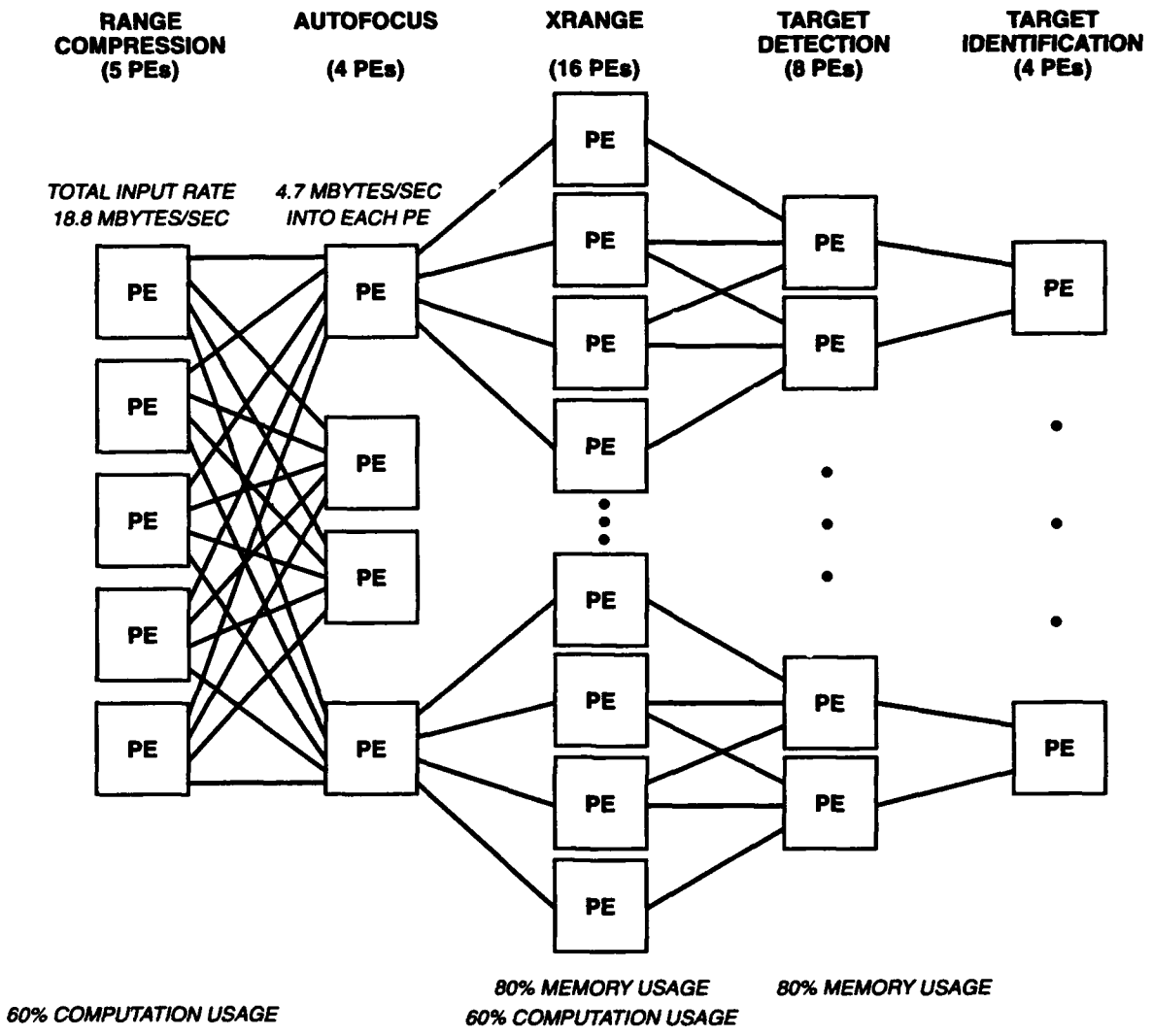

Figure A-1. SAR processor.

A.2.2 Autofocus

The autofocus function has yet to be precisely defined. It is expected to be based upon the phase-gradient algorithm and will include multiple transform passes (FFTs) on the data (perhaps a subset of the data).

It is assumed that the data will be autofocused in 128-pulse batches. Also, while the data from all three polarizations will be passed through the autofocus PEs, only one channel will be used to compute autofocus estimates. Finally, the computation requirement was based upon the equivalent of three 128-point FFTs at each range gate for each 128-pulse batch. This amount of computation power was then doubled to determine that four PEs were required for autofocus.

While the range-processing PEs handle the data in single-pulse batches, the autofocus PEs work on a 128-pulse subframe [one second of data, or about 512 pulses, is referred to as a frame because images are created in one-second batches (synthetic apertures)]. Note that the data is corner-turned implicitly during the transfers from the range to autofocus PEs and then internally within each autofocus PE. As each batch of data for autofocus is only one-quarter of a frame, four PEs can simultaneously maintain an input buffer and an output buffer while processing the subframe in between. Autofocus estimates are transmitted to each cross-range processing PE. Input and output data rates are 4.77 Mbytes/sec for each PE.

A.2.3 Azimuth Processing

Cross-range resolution is achieved by convolving across the pulse dimension at each range gate. This is currently done using fast convolution (i.e., FFT-based) techniques. The radar data is convolved with a Taylor-window-based filter. A quadratic phase correction and the autofocus estimates computed in the previous step are incorporated into the filter. A fairly large number of contiguous range gates are expected to use the same quadratic phase corrections (this saves storage).

The current implementation of this processing works on two frames (1024 pulses) of data. For each of the three polarizations, at each range line, the convolution is computed by a complex multiply sandwiched between two 1024-point FFTs. The appropriate 512 points represent the complex image pixels at the given range for the given polarization. A magnitude operation is performed to create an image for the given polarization and is combined with the complex pixels from the other two polarizations to form a reduced-speckle (e.g., PWF) image. The four images are made available to the detection and identification processing stages.

Sixteen PEs are required to hold the data (given input and output buffering) for the cross-range processing. About sixty per cent of the sixteen PEs' computing power is needed. The input data rate for each PE is about 1.2 Mbytes/sec. The output rate is 0.8 Mbytes/sec/PE. This assumes that the outputs of the cross-range processing are four images per second, one for each polarization and the reduced-speckle image, and that each image consists of 512 (cross-range dimension) by 2048 (range) pixels. Each pixel is assumed to be real (the result of a magnitude operation).

A.3 DETECTION AND IDENTIFICATION

Twelve PEs have been allocated for detection and identification processing. The sizing is driven largely by the requirement to maintain four images in memory. Thus, there is a good deal of computational capability in reserve should algorithms evolve.

One-meter cells are used in the CFAR detection process. A cell is computed by adding up a 4 × 4 collection of high-resolution pixels (from the PWF image). It is assumed that the resulting ground clutter data is lognormally distributed. Thus, the logarithm of each cell is computed, and a two-parameter (mean and variance) test is used to find detections. The perimeter of a 40 cell × 40 cell box centered on the target cell is used to estimate the two parameters. In practice, eight lead-lag vectors must be maintained: four for mean estimation and four for sums-of-squares computation for variance estimation. Each of the four vectors represents a side of the 40 cell × 40 cell box.

Threshold crossings will be clustered; if a detection is within a target's length of a previous detection, the two will be combined to form a cluster. It is envisioned that the cluster location will be computed as a simple amplitude-weighted average of the locations of the detections in both range and cross-range. Mature clusters (clusters far enough from the data being presently processed so that no new detections will be added to them) are used to flag regions of interest.

Some of the operations associated with texture discrimination will be described. At this stage, the data rate has been reduced by the CFAR to (on average) six regions of interest per second. A region of interest (ROI) is, for example, a 175 × 175 pixel area centered upon a detected cluster. The size of the region depends on the target's size and is predicated on the assumption that the center of the region (the location of the detection) could be any point on the target. A square with sides twice the size of the target's longer dimension would be guaranteed to contain the target within it. The following is a description of some of the steps performed upon a region of interest and the target-sized rectangle once its orientation has been estimated.

Orientation involves placing the "target" (e.g., an 86 × 12 rectangle) within the "region of interest" (a 175 × 175 square with the centroid location at its center). Rectangles formed from a sequence of slides (by four or eight pixels) and rotations (3-degree increments) are tested. The slide/rotation position that contains the maximum energy is the assumed location/orientation of the target within the region of interest.

The 50 brightest pixels in the 175 × 175 region are used to estimate (fractal) dimensionality within the region. An algorithm is run to determine the minimum number of 2 × 2 boxes required to "cover" the 50 brightest pixels. Relatively few 2 × 2 boxes means that the bright spots clump together. This is indicative of a dimensionality of 2 and a high likelihood of a target. Many 2 × 2 boxes indicates that the bright spots are spaced randomly. The dimensionality is near zero and a target is not indicated. In-between values may indicate a target (a one-dimensional "line" along the target may have been detected).

The standard deviation of the logarithms of the pixels in the 86 × 12 template is computed. A high standard deviation is a strong indication of a target.

The power in the 25 (say) brightest pixels in the 86 × 12 template is compared to the power in all the pixels. For targets, the brightest scatterers account for a significant portion of the total power.

APPENDIX B
EDSAP SIZING FOR 8- AND 64-CHANNEL PHASED-ARRAY RADARS

This appendix presents an EDSAP sizing for 8- and 64-channel phased-array radars. The process illustrates two key properties of the EDSAP architecture. One is the ease with which one can integrate or interface to commercially available components. The other is the ability to scale the EDSAP to the task at hand. An EDSAP for 8-channel and 64-channel versions of a phased-array radar is described. This appendix was originally written as part of the study that is reported in Shaw, et al. [3]. Since the time of the study, technology has progressed to the point where the EDSAP is expected to run faster than the speed assumed in the study. This appendix uses the same assumptions as the study: an EDSAP with its processing elements running at 20 MIPS (million instructions per second), which corresponds to 60 MOPS (million operations per second), communication channels that can sustain a communication rate in each direction of up to 15 Mbytes/sec, and a chassis that can hold eight 6.5 in. × 12.0 in. boards.

The phased-array application involves adaptive beamforming followed by Doppler filtering for moving target detection. The processing can be broken down into four steps:

1. For each of the channels (elements of the phased array), a finite impulse response (FIR) filter (a) fabricates complex samples from the radar returns, (b) performs pulse compression, and (c) provides channel equalization.
2. A subsample of the data from each channel is used to estimate the adaptive weights required for beamforming. This part of the processing will be referred to as the sampled matrix inversion (SMI).
3. The beamforming step consists of applying the estimated weights to combine the eight (or 64) channels into a single stream of data (single beam).
4. The vector-processing step consists of Doppler processing, clutter cancellation, and constant false alarm rate (CFAR) detection.

B.1 SIZING OF EDSAP FOR 8-CHANNEL PHASED-ARRAY RADAR

The processing is partitioned as follows: FIR filtering, beamforming, vector processing, and sampled matrix inversion (SMI). Figure B-1 shows the processing flow for the system. The numbers given in the figure are the data rates expected over the indicated paths in Mbytes/sec. The data comes from eight A/D converters, each operating at 4.5 million 12-bit real samples/sec or 6.75 Mbytes/sec. Each of these data streams goes into an FIR filter that creates the I and Q samples, equalization, and pulse compression. The output from each of the FIR filters is 0.75 million complex 2 × 16-bit numbers/sec or 3 Mbytes/sec. The output of the FIR filters goes to two different places:

1. Each of the eight channels goes to each of the beamformers, where a weighted sum is taken to derive a beam.
2. Each of the eight channels also interfaces to the SMI processing, where a covariance matrix is estimated and used to compute appropriate weights for the adaptive nulling.

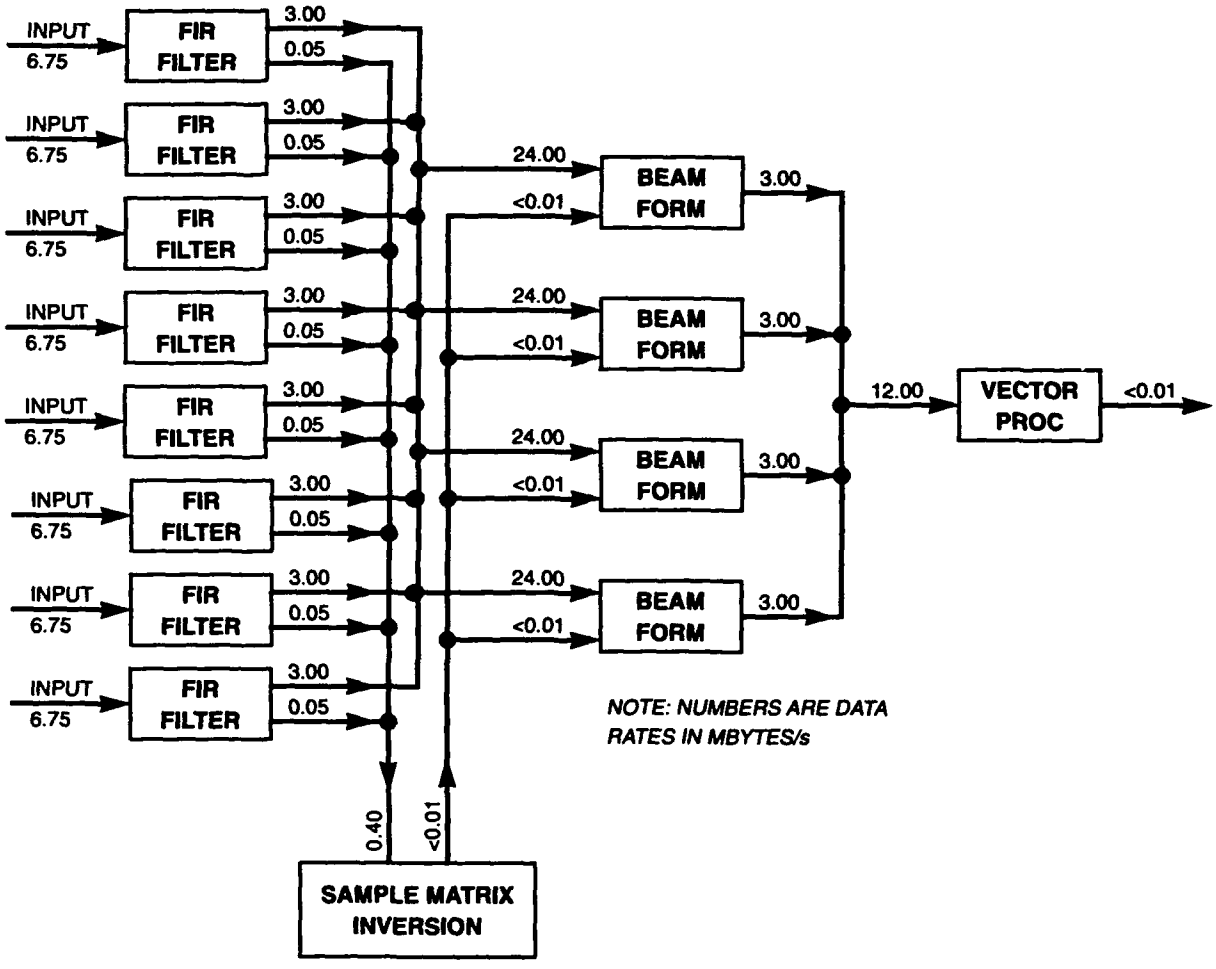

Figure B-1. Processing flow for 8-channel phased-array radar.

The existing implementation has the FIR filters outputting all their data onto a bus. The SMI processor then grabs the small percentage of the samples that it needs from that bus. In the EDSAP implementation, rather than send all the data to the SMI and let it sort out what it needs, each FIR filter sends only the data that is actually to be used by the SMI. The output of the SMI processing is < 0.01 Mbytes/sec to each of the beamformers. The output of the beamformers is 0.75 million complex 2 × 16 bit words/sec, or about 3 Mbytes/sec. The beamformer output goes to the vector processing. The vector processing performs MTI radar processing on each of the beams. The vector processing outputs a very low data rate (< 0.01 Mbytes/sec).

TABLE B-1
Processing and Communication for 8-Channel Radar

Board function	Board type	Qty	Source	Mbytes/s	Destination	Mbytes/s
FIR	Quad FIR	2.00	A/D	54.0	Beamform	24.0
					SMI	0.4
SMI	Hex PE	0.16	FIR	0.4	Beamform	<0.01
Beamform	Hex PE	1.33	FIR	24.0	Vector Proc	12.0
Vector Processor	Hex PE	1.00	Beamform	12.0	Data Proc	<0.01

Table B-1 lists the processing and communication necessary for implementing the 8-channel radar with the EDSAP. The first column is the board function followed by the board type. Next is the number of copies of that board type that are required (these numbers will be justified later in this section). Also shown is the input source for the data into the processing module and the bandwidth of that data in millions of bytes per second, followed by the destination for the data output by the module and its bandwidth. It should be noted that communication within a module is not accounted for in this table.

B.1.1 FIR Filter Implementation

In the current system each of the eight channels contains a FIR filter. These FIR filters are implemented with eight A100 filter ICs and two adders. Each A100 performs approximately 96 million multiply accumulates per second. The processing element (PE) performs one multiply and two additions per instruction cycle. Running at a 20-MIPS rate, a PE can perform 20 million multiply accumulates per second. Thus, it would take 40 PEs to replace the eight A100s currently being used to implement the filter. It was therefore decided to continue using the A100s to implement the FIR filters.

The EDSAP PE can be used as a general-purpose programmable interface chip. In this case it is used to interface the FIR filter to the rest of the EDSAP. Figure B-2 is a diagram for a quad (containing four FIR filters) FIR filter board that is the same size as a hex PE board; it can be plugged into a hex PE slot within the EDSAP.

B.1.2 Sampled Matrix Inversion

The samples selected for the SMI processing are determined by the PE chips used to interface the FIR filters.

The SMI process uses a relatively small number of input samples to estimate the weights for adaptive beamforming. In the current processing scheme, this function requires approximately five MOPS (implemented using a single Motorola DSP56001). This indicates that the function could easily be performed in a single PE (60 MOPS) of a hex PE board. The beamforming weights produced must be applied

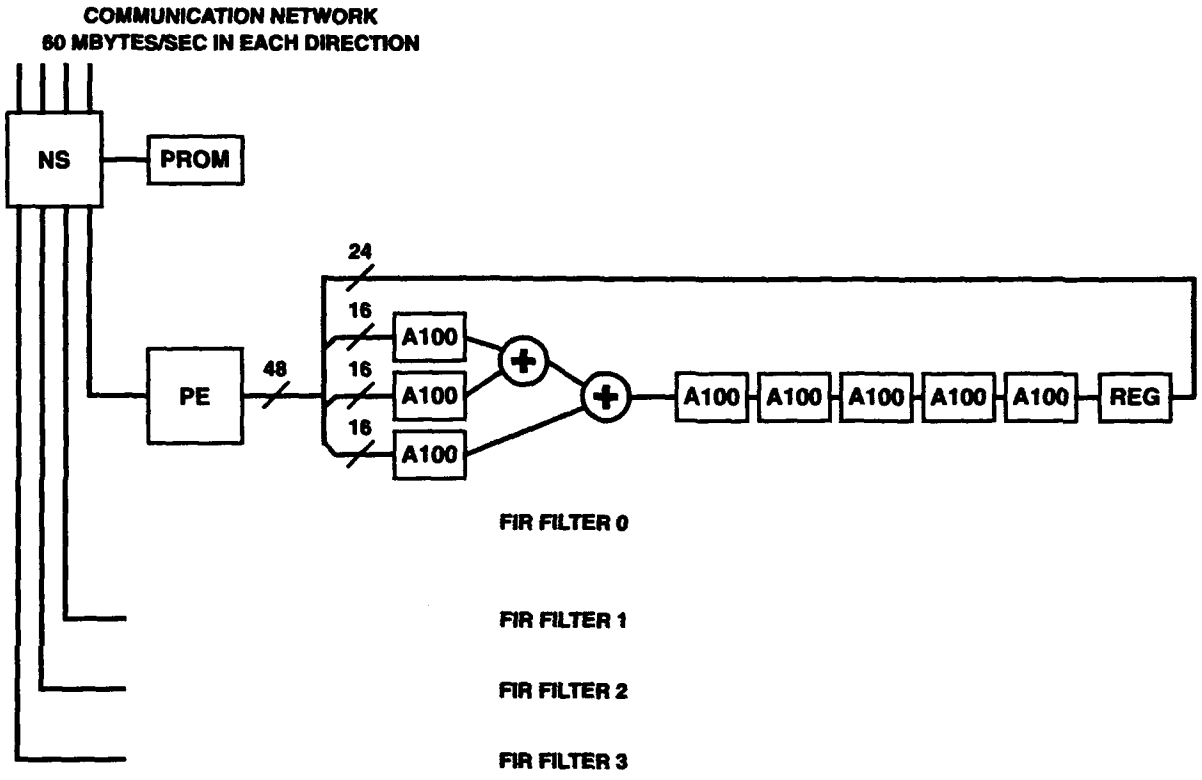

Figure B-2. Quad FIR filter board.

to data taken at the same time as the data used to estimate those weights. This means that the eight-channel input to the beamforming function must be stored (buffered) during the time it takes to estimate the weights. It is expected that the 10-msec lag associated with the present SMI processing scheme is a reasonable estimate of the latency for this same processing on a single PE.

The adaptive nulling weights derived by SMI would be applied in the beamforming section. The beamforming section would buffer approximately 10 msec worth of data so that weights could be applied to the data they were derived from.

B.1.3 Beamforming

Beamforming requires 24 million multiply/accumulates per second per beam. Each PE on a hex PE card is capable of 20 million multiplies and 40 million additions per second. Therefore two PEs (or one third of a hex PE) should be able to perform the beamforming for one beam with margin. To process four beams, eight single PEs or $1\frac{1}{3}$ hex PEs will be allocated.

The beamforming PEs would buffer the data, delaying it enough so weights that correspond to the appropriate samples of the SMI can be applied to the data.

B.1.4 Vector Processing

In the current system the vector processing is done by four vector processors. Each of these processors currently employs an array of eight Motorola DSP56001 DSP chips and a bus-oriented communication network.

The ability of the PE architecture and instruction set to efficiently perform Doppler filtering and CFAR was demonstrated with the DSAP processor in the UAV radar [2]. A comparison of the performance of the DSAP in the UAV radar with the vector-processing requirements of this system indicates that a single hex PE would be sufficient.

B.1.5 Communication Network

Figure B-3 is a block diagram of the EDSAP sized to implement an 8-channel phased-array radar; the shaded area is the backplane. The communication network is part of the backplane (the EDSAP has an active backplane). The system shown in Figure B-3 includes the card complement specified by Table B-1. The quad FIR and hex PE cards indicated by dotted lines are optional spares and are included to give a degree of fault tolerance.

B.2 SIZING OF EDSAP FOR 64-CHANNEL PHASED-ARRAY RADAR

The 64-channel system is essentially the same as the 8-channel system except that a weighted sum of 64 channels is being taken for each of four beams that are formed. This also means that the SMI processing must get data from 64 channels instead of eight.

B.2.1 FIR Filter and Beamforming

The same EDSAP quad FIR filter board proposed for the 8-channel system can be used for the 64-channel system. Hex PE cards are used for beamforming. Figure B-4 shows a way to use a quad FIR filter card and a hex PE to perform the FIR filtering and beamforming on four input channels for all four beams. The numbers shown for each of the data paths specify the data rate in Mbytes/sec. For simplicity, Figure B-4 does not show all the data paths in and out of PEs 1, 2 and 3. Basically, PEs 0 through 3 cooperate to start the formation of four beams for four input channels. Each PE applies the four sets of weights (one for each beam) to the input stream from one channel. The PEs then shuffle the data so that each PE has the four weighted samples required for starting the formation of one beam. These samples are summed and output to other PEs that will finish the formation of the 64-element beam. This will now be described in greater detail.

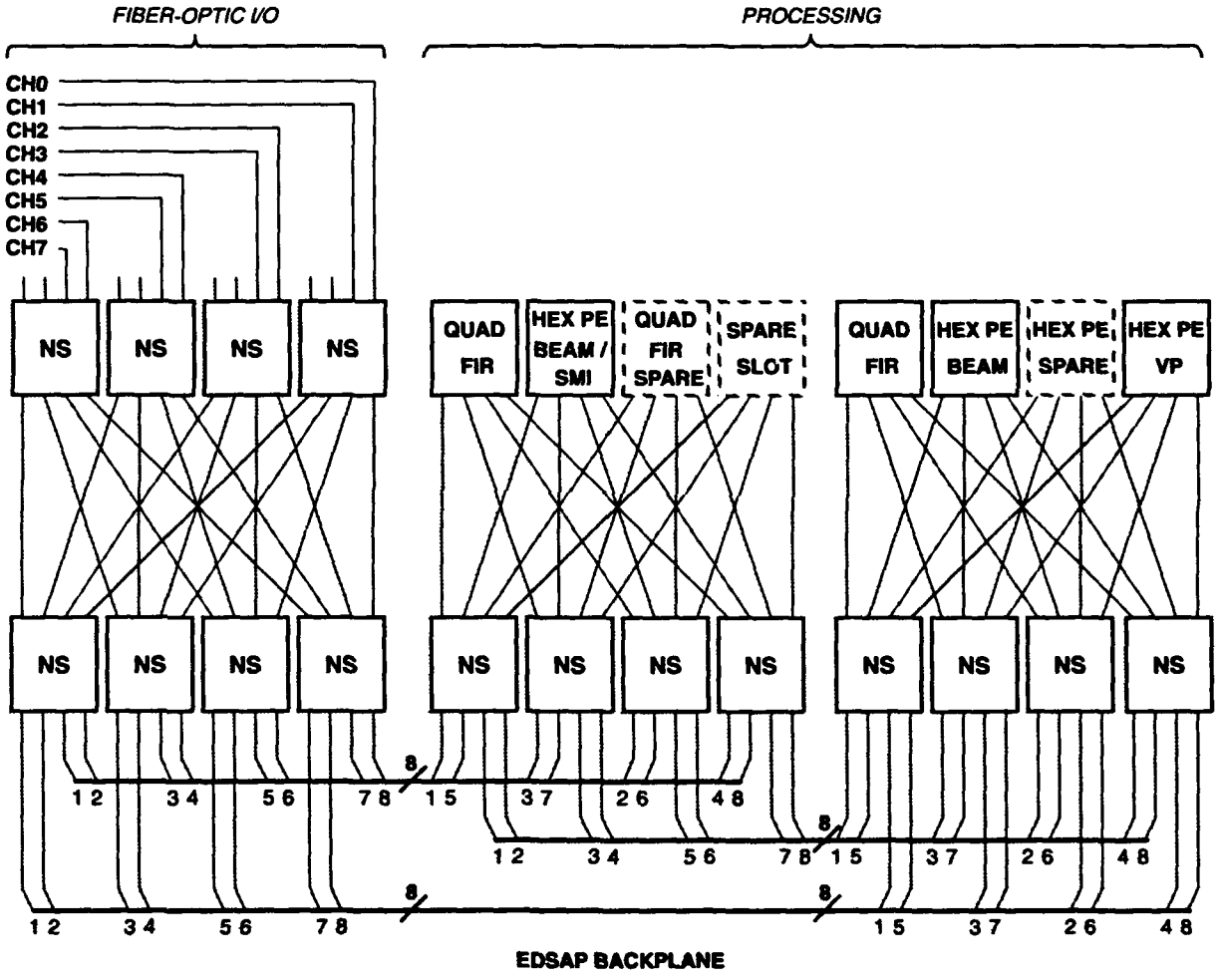

Figure B-3. EDSAP topology for implementing an 8-channel phased-array radar.

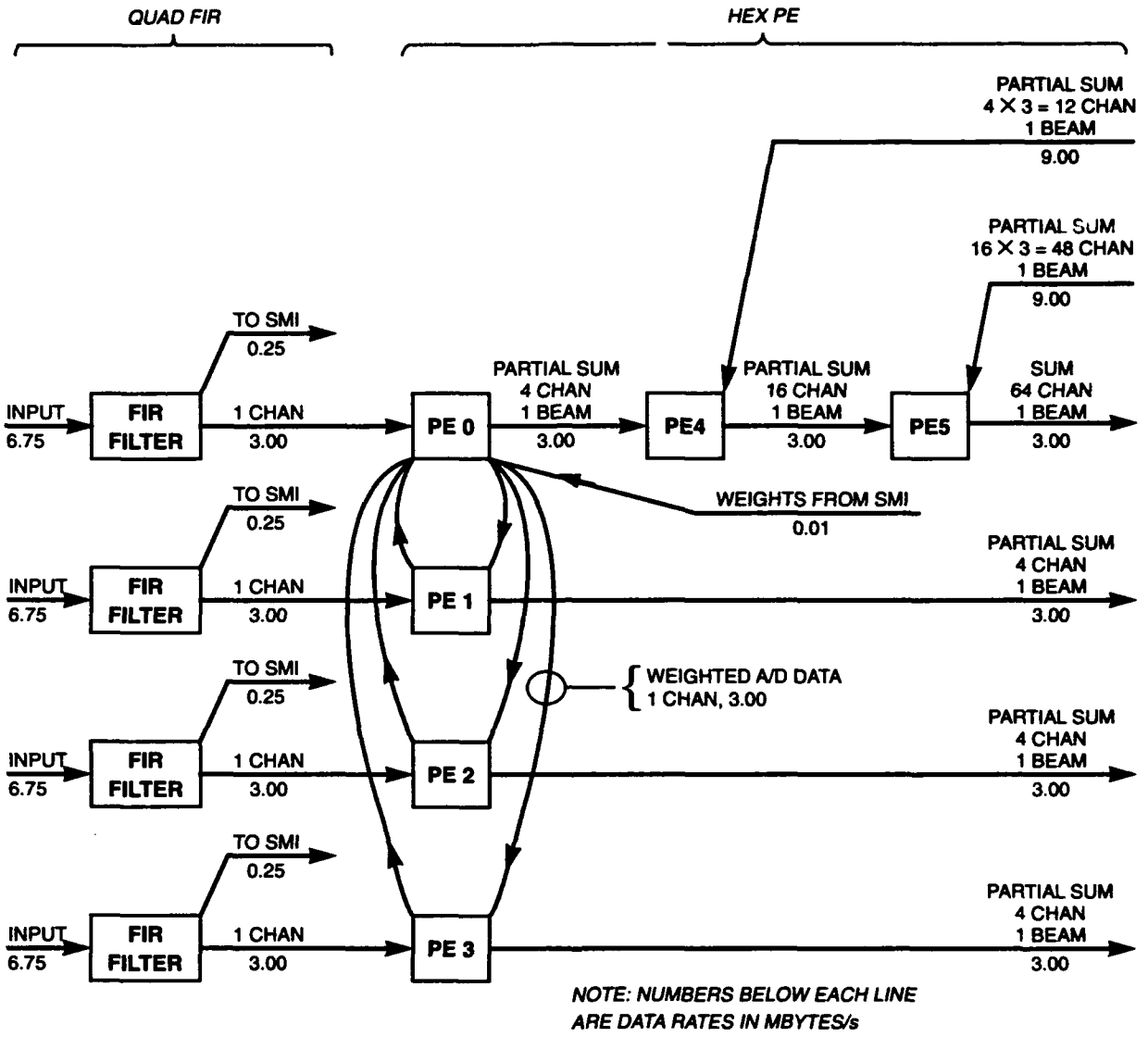

Figure B-4. EDSAP FIR filter and beamforming data flow.

PE 0 performs its processing steps concurrently on successive sets of data. That is, it applies the complex weights to the input stream at the same time it forms the appropriate beam using the most recently shuffled data. These same steps are performed by PEs 1, 2 and 3, although the explanation that follows is from PE 0's point of view. The steps to process each batch of data are as follows:

1. Complex data words (16 bits I, 16 bits Q) are output from the FIR filter at a rate of 0.75 MHz, corresponding to a data rate of 3.00 Mbytes/sec. This PE input data is buffered until PE 0 receives processing weights from the SMI processing.
2. Once PE 0 receives one weight for each of the four beams from the SMI processor, PE 0 applies the weights to its data, creating partial sums for all four beams. This processing requires 0.75 million complex multiplies per second per beam for a total of 12 million real multiplies and six million real additions per second for all four beams.
3. The weighted data for beam 0 is kept in PE 0. The weighted data for beams 1, 2 and 3 is sent on to PEs 1, 2 and 3 respectively. This weighted data forms three streams each at 0.75 million complex words/sec or 3.00 Mbytes/sec. Each of these streams contains data corresponding to PE 0's input channel, weighted for a particular beam. While PE 0 is sending the three steams of data to PEs 1, 2 and 3, PEs 1, 2 and 3 are sending PE 0 their data weighted for beam 0.
4. PE 0 takes the weighted beam 0 data from PEs 1, 2 and 3 and adds it to its own. This requires 2.25 complex or 4.5 million real additions per second.
5. The partial sum of four input channels for beam 0 is sent to PE 4 by PE 0.

PE 4 adds the beam 0 partial sum from PE 0 to those from three other PEs (on other hex PE boards), creating a partial sum accumulated from the data of 16 A/D input channels. This requires 2.25 complex or 4.5 million real additions per second.

Ignoring for a moment what PE 5 is doing, it is easy to see that four quad FIR filters and four hex PEs can be used to form a cluster that derives partial sums of 16 input channels for all four beams. Four of these clusters can be combined to process 64 channels. One PE 5 in each of these clusters (the other three are spares) can be used to take the 16 partial sums and create the sum for all 64 channels for a particular beam. Hence, four beams have been formed from 64 input channels.

It is interesting to note that the amount of beamforming a PE can perform is limited by its I/O to the communication net (not the bandwidth of the communication net) rather than by processing rate. The arithmetic processor (AP) within the PE has two ALUs capable of performing a total of 40 million real additions and 20 million real multiplies per second. PEs 0, 1, 2 and 3 are each performing 10.5 million real additions and 12 million real multiplies per second. PEs 4 and 5 are not as busy. However, the communication input to all the PEs used for the beamforming is just over 12 Mbytes/sec of the available 15 Mbytes/sec.

B.2.2 Sampled Matrix Inversion

The samples for SMI processing are determined by the PEs used to interface the FIR filters.

The weights derived by SMI are applied in the beamforming section. The beamforming section buffers approximately seven msec worth of data so that weights can be applied to the data they were derived from.

The SMI task for the 64 channels is much greater than it is for the 8-channel system. A special wafer-scale device called MUSE [8] has been developed to perform the SMI processing for the 64-channel system. The total data rate into the MUSE device is somewhere around 12 Mbytes/sec (16 Mbytes/sec was budgeted in sizing the communication net) and the rate out is around 50 Kbytes/sec. The MUSE is interfaced with two PEs for input and one PE for output, similar to the way the A100s are interfaced on the FIR filter board.

B.2.3 Vector Processing

The vector processing for the 64-channel system is analogous to that of the 8-channel system, as both systems form four beams after interference cancellation. A single hex PE is again used to implement the vector processing (VP) for all four beams.

B.2.4 Network and Physical Layout

The EDSAP for the 64-channel system is divided into two superclusters, each contained in its own chassis. The superclusters are in turn made up of smaller clusters. Figure B-5 shows the cluster used for FIR filtering and beamforming of 16 input channels. This cluster contains four pairs of quad FIR filters and hex PEs that plug into the backplane of the supercluster. The backplane contains 12 network switches to achieve the necessary network bandwidth. There are 16 bidirectional connections for the A/Ds and eight to connect to the rest of the supercluster.

Figure B-6 is a diagram of a supercluster, which is made up of the following:

1. Two 16-channel FIR/beamforming clusters, along with their 32 bidirectional fiber-optic connections for the A/Ds (total of 64 fibers).
2. A spare quad FIR filter.
3. A spare hex PE that can be used for either beamforming or vector processing.
4. A MUSE board. There is a MUSE board in each of the superclusters, only one of which is actually used in normal operation. The other is a spare for fault tolerance. It is possible to run the MUSE boards in both superclusters and compare the results as a diagnostic.
5. A hex PE that may be used as a VP. The VP in either supercluster is adequate to handle the required vector-processing job; the other becomes a spare. The spare VP can be used for beamforming as well as vector processing.
6. Sixteen bidirectional fiber-optic I/O ports, eight of which are used to tie the two superclusters together. The other eight are available to devote to other devices, possibly spare A/Ds.

An EDSAP supercluster for 32 channels of the 64 would require twenty-four 6.5 in. × 12 in. boards spaced at 0.75 in. The communication network is part of the backplane. The fiber-optic transmitters/receivers could be contained either on a separate board or on the backplane. The whole supercluster should fit in a chassis 8 in. high × 15 in. wide × 22 in. deep (1.5 ft^3). Assuming 20 W per board, 1 W for each of the 36 NSs on the backplane, 20 W for clock generation/distribution, 2 W for each of the 48 fiber-optic ports, and

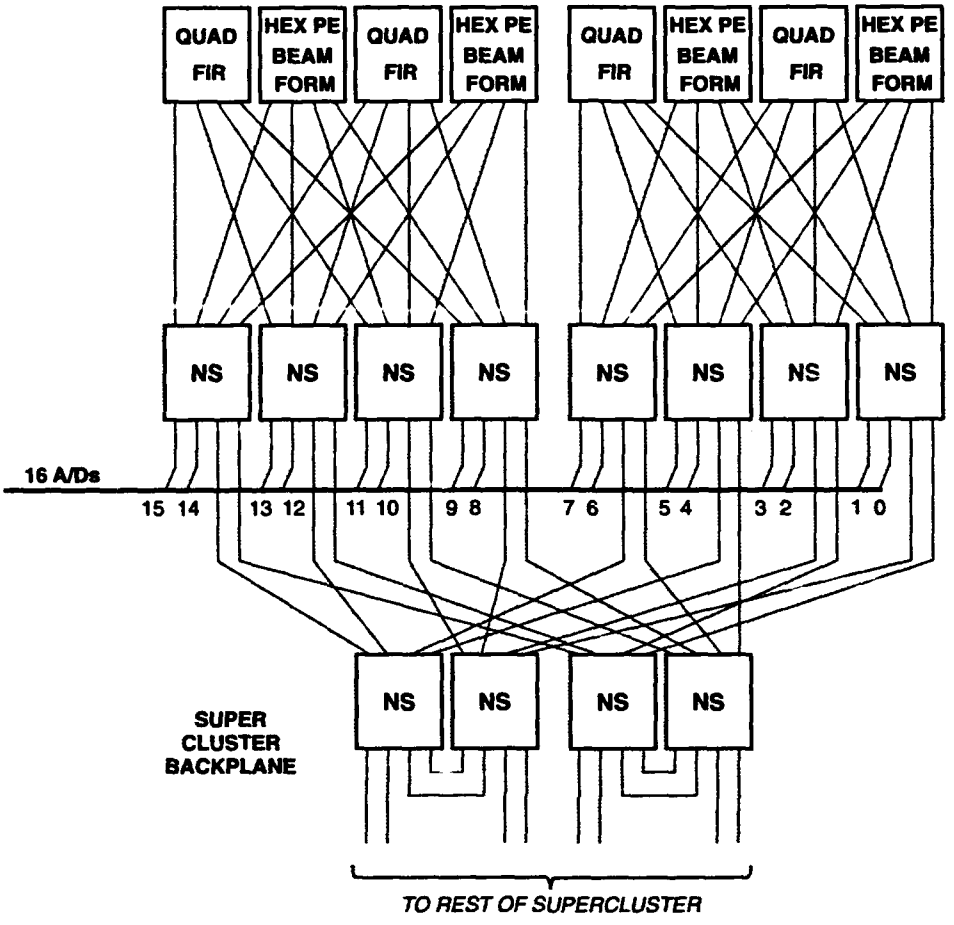

Figure B-5. EDSAP 16-channel FIR filter and beamforming cluster.

power supplies of 80% efficiency, the total power dissipated by each of the two superclusters should conservatively be under 800 W.

In order to demonstrate that the communication network can handle the data rates, Table B-2 shows some representative data rates through the areas that are most likely to present bottlenecks. Note the spare bandwidth available; clearly, bandwidth is more that adequate. The unused bandwidth allows the spare boards to serve either of the superclusters.

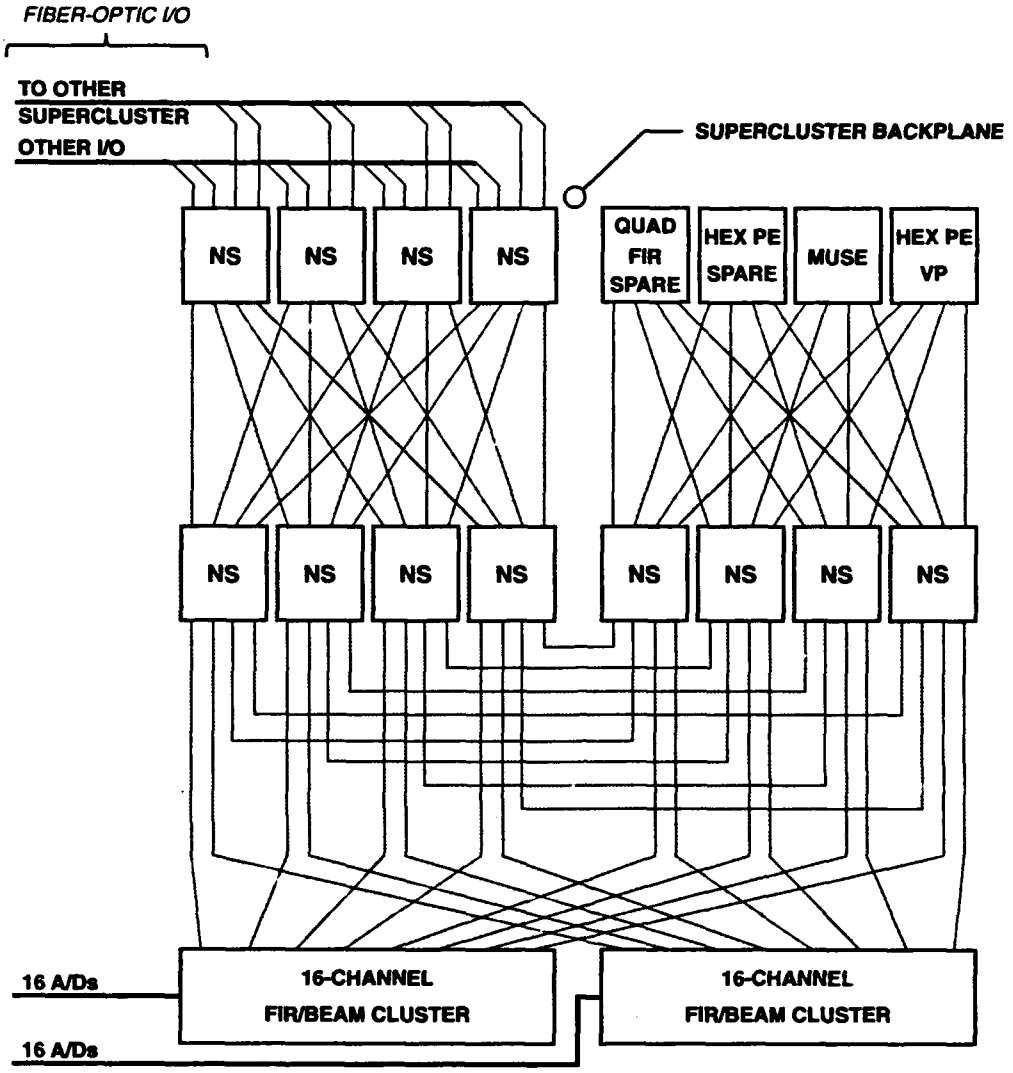

Figure B-6. EDSAP 32-channel supercluster.

TABLE B-2
Communication Data Rates for 64-Channel System

Data Source and Operations	MBytes/sec
Data out of 16-channel FIR/beamforming cluster:*	
1 Beam output from beamforming	3.00
Partial sums of 16 channels for 3 beams (3 × 3.00)	9.00
Samples from FIR filters to SMI (16 × 0.25)	4.00
Total data rate out of cluster (out of available 120)	16.00
Data into FIR/beamforming cluster from rest of system:*	
Partial sums of 16 channels for 1 beam (3 × 3.00)	9.00
Weights from SMI processing to beamformers (16 × 0.01)	0.16
Total data rate into cluster (out of available 120)	9.16
Data into supercluster doing MUSE and VP for both superclusters:	
2 Beams output from beamforming	6.00
Partial sums of 16 channels for 2 beams	6.00
Samples from FIR filters to SMI (32 × 0.25)	8.00
Total data rate into supercluster (out of available 120)	20.00

* Does not include the I/O for the A/Ds

REFERENCES

1. F.E. Hall and A.G. Rocco Jr., "A compact programmable array processor," *The Lincoln Laboratory Journal* Vol. 2, No. 1, 41–62 (1989).

2. C.E. Schwartz, T.G. Bryant, J.H. Cosgrove, G.B. Morse, and J.K. Noonan, "A radar for unmanned air vehicles," *The Lincoln Laboratory Journal* Vol. 3, No. 1, 119–143 (1990).

3. G.A. Shaw, R.A. Gabel, D.R. Martinez, A.G. Rocco, S.C. Pohlig, A.D. Gerber, J.K. Noonan, and K. Teitelbaum, "Multiprocessors for radar signal processing," MIT Lincoln Laboratory, Lexington, Mass., Technical Report TR-961 (17 November 1992).

4. "Testability bus specification," IEEE draft specification P1149/D7 (31 October 1988).

5. Q.L. Klein, private communication (22 December 1988).

6. P. Rygiel, private communication (29 April 1988).

7. Q.L. Klein, private communication (22 December 1988).

8. C.M. Rader, "Wafer-scale integration of a large systolic array for adaptive nulling," *The Lincoln Laboratory Journal* Vol. 4, No. 1, 3–30 (1991).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	6 May 1993	Technical Report

4. TITLE AND SUBTITLE

Overview of Enhanced Data Stream Array Processor (EDSAP)

5. FUNDING NUMBERS

C — F19628-90-C-0002
PE — 63250F
PR — 224

6. AUTHOR(S)

A. Gregory Rocco, Paul D. Linton, and James K. Noonan

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lincoln Laboratory, MIT
P.O. Box 73
Lexington, MA 02173-9108

8. PERFORMING ORGANIZATION REPORT NUMBER

TR-974

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Air Force
Electronic Systems Center
Hanscom AFB, MA

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ESC-TR-92-159

11. SUPPLEMENTARY NOTES

None

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

The Enhanced Data Stream Array Processor (EDSAP) is a scalable version of the DSAP processing architecture that will allow massively parallel processing. Lincoln Laboratory previously designed two generations of DSAP processors; these programmable signal/data processors provide very high throughput in small, lightweight packages that require low power use. The EDSAP design builds upon the DSAP architecture in a way that provides for processing in the gigaop (billion operations per second) to teraop (trillion operations per second) range. This capability is being accomplished primarily by a significant redesign of high-speed I/O and interprocesso· communication. Other architectural improvements, as well as rapid technological advances, contribute to the improved processor. This report provides an overview of the architecture of the EDSAP. The first EDSAP board is expected to be available in late 1993.

14. SUBJECT TERMS

synthetic aperture radar; phased array radar; radar signal processing; parallel processing; real-time processing; packet-switched network; digital signal processing; communication network(s); massively parallel processing

15. NUMBER OF PAGES
64

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Same as Report